THIS MANUAL IS CURRENT
FOR MDBS 3.09

MDBS III

DATA MANAGEMENT SYSTEM

APPLICATION PROGRAMMING REFERENCE MANUAL

VERSION 3.09

MARCH 1987

Micro Data Base Systems, Inc.

P. O. Box 248

Lafayette, Indiana 47902

USA

Telex: 5106017487 (MDBS LAF UQ)

(800) 344-5832

Copyright Notice

This entire manual is provided for the use of the customer and the customer's employees. The entire contents have been copyrighted by Micro Data Base Systems, Inc., and reproduction by any means is prohibited except as permitted in a written agreement with Micro Data Base Systems, Inc.

© Copyright 1987 Micro Data Base Systems, Inc.

NEW VERSIONS, LICENSING, AND DISCLAIMER

Any programming endeavor of the magnitude of the MDBS III software continually evolves over time. Realizing this, Micro Data Base Systems, Inc. (**mdbs**), provides updates to this product as part of a service and maintenance program.

New versions of MDBS III software are considered separate products; however, bona fide owners of previous versions are generally entitled to a preferential rate structure.

Each copy of our software is personalized to identify the licensee. There are several levels of this personalization, some of which involve encryption methods guaranteed to be combinatorially difficult to decipher. Our products have been produced with a very substantial investment of capital and labor, to say nothing of the years of prior involvement in the data base management area by our principals. Accordingly, we are seriously concerned about any unauthorized copying of our products and will take any and all available legal action against illegal copying or distribution of our products.

All rights reserved. No part of this material shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without written permission from **mdbs**.

Although care has been taken in the preparation of this material, **mdbs** assumes neither responsibility for errors or omissions, nor for damages resulting from the use of the information contained herein. **mdbs** does not warrant its accuracy nor guarantee the operation of the system in every instance described herein.

The reader/user assumes full liability and all risk for any damages resulting from the use of the information contained herein, and for determining whether the information contained herein is suitable for user's intended purpose.

mdbs reserves the right to incorporate design improvements and new functions in its software products and software systems. Recent improvements may not always be reflected in documentation.

© Copyright 1987 Micro Data Base Systems, Inc.

PREFACE

This manual provides an introductory overview of application development with MDBS III. It then concentrates in detail on the features and utilization of the MDBS III Data Description Language. Companion manuals discuss the MDBS III Data Manipulation Language and various optional modules. For a tutorial treatment of data base management (explaining its value, characteristics, and advantages), consult such suitable references as:

- Micro Database Management - Practical Techniques for Application Development by R.H. Bonczek, et. al., 536 pages, Academic Press, New York, 1984.
- "A Perspective on Data Models," PC Tech Journal, Vol. 2, No. 1, 1984.
- "Micros Get Mainframe Data Scheme," Systems and Software, Vol. 3, No. 5, 1984.
- "Uniting Relational and Postrelational Database Management Tools", Systems and Software, Vol. 3, No. 11, 1984.

The first reference provides the most definitive coverage of the postrelational approach, comparing and contrasting it with the four older data base management approaches. That book also includes extensive examples of MDBS III usage. It is available through your local bookstore, from the publisher, or from MDBS, Inc.

The MDBS commitment to customer success does not stop with software innovation, quality and support. The company offers a full range of consulting and application development services to clients from regional offices located throughout the United States and Canada. Here are some of the services MDBS provides:

- Customized application systems design
- Customized application systems development using GURU, KnowledgeMan and/or the MDBS III tools
- Communications interfaces including mainframe-micro links
- Training seminars for MDBS III, KnowledgeMan, and GURU in major cities and at customer sites
- Special training on a wide variety of topics including specific application systems, C programming, MDBS III usage, and GURU usage
- General consulting including feasibility studies and hardware/software recommendations

MDBS, KnowledgeMan, and GURU are registered trademarks of Micro Data Base Systems, Incorporated.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Table of Contents

TABLE OF CONTENTS

Page

PREFACE v

CHAPTER I. OVERVIEW I-1
The Role of DML in Application Development I-2

CHAPTER II. GENERAL BACKGROUND II-1
Data Transferral II-1
Currency Indicators II-2
Stating a DML Command II-3
Multiuser Environment II-6
Protecting Data Base Consistency against External Factors II-7

CHAPTER III. FIND COMMANDS III-1
Command Details III-3

CHAPTER IV. RETRIEVAL COMMANDS IV-1
Command Details IV-3

CHAPTER V. MODIFY COMMANDS V-1
Command Details V-3

CHAPTER VI. ASSIGNMENT COMMANDS VI-1
Command Details VI-5

CHAPTER VII. CREATION COMMANDS VII-1
Command Details VII-2

CHAPTER VIII. CONNECT COMMANDS VIII-1
Command Details VIII-2

CHAPTER IX. DISCONNECT COMMANDS IX-1
Command Details IX-2

CHAPTER X. DELETION COMMANDS X-1
Command Details X-2

CHAPTER XI. SCHEMA ACCESS COMMANDS XI-1
Command Details XI-1

CHAPTER XII. UTILITY COMMANDS XII-1
Command Details XII-2

© COPYRIGHT 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Table of Contents

Page

CHAPTER XIII. BOOLEAN COMMANDS XIII-1
Command Details XIII-3

CHAPTER XIV. SPECIAL COMMANDS XIV-1
Command Details XIV-2

CHAPTER XV. MULTIUSER LOCKING COMMANDS XV-1
Command Details XV-6

CHAPTER XVI. RECOVERY COMMANDS XVI-1
Command Details XVI-3

CHAPTER XVII. COMMAND STATUS DESCRIPTIONS XVII-1

APPENDIX A: DML COMMAND FORMATS A-1

Figure VI-1. Example Schema Diagram VI-3

TABLE XI-1. Data Structure for GAI Command XI-2
TABLE XI-2. Data Structure for GII Command XI-7
TABLE XI-3. Data Structure for GKI Command XI-10
TABLE XI-4. Data Structure for GRI Command XI-14
TABLE XI-5. Data Structure for GSI Command XI-17

TABLE XV-1. Multiuser Locking Privileges XV-3
TABLE XV-2. Data Variable Structure for MRID Command XV-10

INDEX I-1

© COPYRIGHT 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Classifications

Command Classifications

Page

ASSIGNMENT COMMANDS

SCD Set Current of run unit to Data base key VI–5
SCM Set Current of run unit based on Member VI–6
SCN Set Current of run unit to Null VI–7
SCO Set Current of run unit based on Owner VI–8
SCU Set Current of run unit based on User indicator VI–9
SDC Save Data base key for Current of run unit VI–10
SMC Set Member based on Current of run unit VI–11
SME Set Member to current of run unit (Exception) VI–12
SMM Set Member based on Member VI–13
SMN Set Member to Null VI–14
SMO Set Member based on Owner VI–15
SMU Set Member based on User indicator VI–16
SOC Set Owner based on Current of run unit VI–17
SOE Set Owner to current of run unit (Exception) VI–18
SOM Set Owner based on Member VI–19
SON Set Owner to Null VI–20
SOO Set Owner based on Owner VI–21
SOU Set Owner based on User indicator VI–22
SUC Set User indicator to Current of run unit VI–23
SUM Set User indicator to Member VI–24
SUN Set User indicator to Null VI–25
SUO Set User indicator to Owner VI–26
SUU Set User indicator to User indicator VI–27

BOOLEAN COMMANDS

AMM And of Members with Members XIII–3
AMO And of Members with Owners XIII–4
AOM And of Owners with Members XIII–5
AOO And of Owners with Owners XIII–6
XMM eXclude Members from Members XIII–7
XMO eXclude Members from Owners XIII–8
XOM eXclude Owners from Members XIII–9
XOO eXclude Owners from Owners XIII–10

CONNECT COMMANDS

IMS Insert Member into Set VIII–2
IOS Insert Owner into Set VIII–3

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Classifications

Page

CREATION COMMANDS

CRA Create Record in Area VII–2
CRS Create Record and Store VII–4

DELETION COMMANDS

DRC Delete Record that is Current X–2
DRM Delete Record that is Member X–3
DRO Delete Record that is Owner X–4

DISCONNECT COMMANDS

RMS Remove Member from Set IX–2
ROS Remove Owner from Set IX–3
RSM Remove all Set Members IX–4
RSO Remove all Set Owners IX–5

FIND COMMANDS

FDRK Find Duplicate Record based on calc Key III–3
FFM Find First Member III–4
FFO Find First Owner III–5
FFS Find First Sequential record III–6
FLM Find Last Member III–7
FLO Find Last Owner III–8
FMI Find Member based on data Item III–9
FMSK Find Member based on Sort Key III–10
FNM Find Next Member III–11
FNMI Find Next Member based on data Item III–12
FNMSK Find Next Member based on Sort Key III–13
FNO Find Next Owner III–14
FNOI Find Next Owner based on data Item III–15
FNOSK Find Next Owner based on Sort Key III–16
FNS Find Next Sequential record III–17
FOI Find Owner based on data Item III–18
FOSK Find Owner based on Sort Key III–19
FPM Find Prior Member III–20
FPMI Find Prior Member based on data Item III–21
FPMSK Find Prior Member based on Sort Key III–22
FPO Find Prior Owner III–23
FPOI Find Prior Owner based on data Item III–24
FPOSK Find Prior Owner based on Sort Key III–25
FRK Find Record based on calc Key III–26

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Classifications

Page

MODIFY COMMANDS

PFC	Put data into Field of Current of run unit	V–3
PFM	Put data into Field of Member	V–4
PFO	Put data into Field of Owner	V–5
PUTC	PUT data into Current of run unit	V–6
PUTM	PUT data into Member	V–7
PUTO	PUT data into Owner	V–8

MULTIUSER LOCKING COMMANDS

MAU	Multiuser Active User indicators	XV–6
MCC	Multiuser Contention Count	XV–7
MCF	Multiuser Current of run unit Free	XV–8
MCP	Multiuser Current of run unit Protect	XV–9
MRID	Multiuser Run unit IDentification	XV–10
MRTF	Multiuser Record Type Free	XV–11
MRTP	Multiuser Record Type Protect	XV–12
MSF	Multiuser Set Free	XV–13
MSP	Multiuser Set Protect	XV–14

RECOVERY COMMANDS

LGFILE	LoG FILE specification	XVI–3
LGFLSH	LoG file buffer FLuSH	XVI–4
LGMSG	LoG file MeSsaGe	XVI–5
PIFD	Pre–Image File Declaration	XVI–6
TRABT	TRansaction ABoRT	XVI–7
TRBGN	TRansaction BeGiN	XVI–8
TRCOM	TRransaction COMmit	XVI–10

RETRIEVAL COMMANDS

GETC	GET data from Current of run unit	IV–3
GETM	GET data from Member	IV–4
GETO	GET data from Owner	IV–5
GFC	Get Field from Current of run unit	IV–6
GFM	Get Field from Member	IV–7
GFO	Get Field from Owner	IV–8
ODRK	Obtain Duplicate Record based on calc Key	IV–9
OFM	Obtain First Member	IV–10
OFO	Obtain First Owner	IV–11

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Classifications

Page

RETRIEVAL COMMANDS (Cont.)

OLM Obtain Last Member IV–12
OLO Obtain Last Owner IV–13
OMI Obtain Member based on data Item IV–14
OMSK Obtain Member based on Sort Key IV–15
ONM Obtain Next Member IV–16
ONMI Obtain Next Member based on data Item IV–17
ONMSK Obtain Next Member based on Sort Key IV–18
ONO Obtain Next Owner IV–19
ONOI Obtain Next Owner based on data Item IV–20
ONOSK Obtain Next Owner based on Sort Key IV–21
OOI Obtain Owner based on data Item IV–22
OOSK Obtain Owner based on Sort Key IV–23
OPM Obtain Prior Member IV–24
OPMI Obtain Prior Member based on data Item IV–25
OPMSK Obtain Prior Member based on Sort Key IV–26
OPO Obtain Prior Owner IV–27
OPOI Obtain Prior Owner based on data Item IV–28
OPOSK Obtain Prior Owner based on Sort Key IV–29
ORK Obtain Record based on calc Key IV–30

SCHEMA ACCESS COMMANDS

GAFN Get Area File Name XI–1
GAI Get Area Information XI–2
GDC Get Design Counts XI–4
GHRV Get High Range Values XI–6
GII Get data Item Information XI–7
GKI Get Key Information XI–10
GLRV Get Low Range Value XI–13
GRI Get Record type Information XI–14
GSI Get Set Information XI–17

SPECIAL COMMANDS

ALTEOS ALTer End Of Set XIV–2
DBINIT Data Base control system INITialization XIV–3
DBLDR Data Base LoaDeR XIV–4
DBSEL Data Base SELection XIV–5
DEFINE DEFINE data block XIV–6
DMSSJP DMS Set JumP XIV–7
EXTEND EXTEND data block XIV–8

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Classifications

Page

SPECIAL COMMANDS (Cont.)

MPL Multiuser Priority Level XIV–9
SETPBF SET Page BuFfer region XIV–10
UNDEF UNDEFine data blocks XIV–11
VARCMD VARiable for CoMmanD string XIV–12
VARCS VARiable for Command Status XIV–13

UTILITY COMMANDS

AUI Allocate User Indicators XII–2
CCU Check Current of run unit against User indicator XII–3
DBCLS Data Base CLoSe XII–4
DBCLSA Data Base CLoSe for Area XII–5
DBCNV Data Base format CoNVersion XII–6
DBENV Data Base ENVironment XII–8
DBOPN Data Base OPeN XII–10
DBOPNA Data Base OPeN Area XII–12
DBSAVE Data Base SAVE XII–13
DBSTAT Data Base STATistics XII–14
GMC Get Member Count XII–15
GOC Get Owner Count XII–16
GTC Get Type of Current of run unit XII–17
GTM Get Type of Member XII–18
GTO Get Type of Owner XII–19
NCI Null all Currency Indicators XII–20
TCN Test Current of run unit for Null XII–21
TCT Test Current of run unit Type XII–22
TMN Test Member for Null XII–23
TMT Test Member Type XII–24
TON Test Owner for Null XII–25
TOT Test Owner Type XII–26
TUN Test User indicator for Null XII–27

© Copyright 1987 Micro Data Base Systems, Inc.

I. OVERVIEW

This is the MDBS DMS Manual, the second component in the set of **MDBS Reference Manuals.** Before using this manual, you should acquaint yourself with the data description facilities presented in the MDBS DDL Manual.

This manual describes the MDBS III Data Manipulation Language (DML) and the interaction occurring between an application program using DML commands and the data base control system (MDBS.DMS). The documentation in this manual is applicable to MDBS III versions 3a and 3c. The DML command actions are independent of host programming languages selected to interface with MDBS.DMS. Characteristics dependent upon a host programming language (such as the command syntax) are described in appropriate **MDBS System Specific Manuals.**

This chapter outlines the role of MDBS.DMS software and its associated Data Manipulation Language (DML) in application development.

Chapter two provides the background to effectively use the MDBS Data Manipulation Language. The chapter discusses the following concepts:

- Data transfer mechanisms
- Data Base currency indicators
- DML command components
- Multiuser environment
- Data Base maintenance and catastrophe prevention

Chapters three through sixteen describe the fourteen classes of DML commands. Each chapter initially discusses the concepts employed by a class of DML commands and identifies the individual commands within the class. Following the general discussion, the chapter presents each command in detail. Each command appears alphabetically with a synopsis of its function and its effects upon the data base environment.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Overview

The DML command classes appear in the following chapters:

DML Command Class	Chapter
Finding records	Three
Retrieving data from records	Four
Modifying data in records	Five
Assigning currency indicators	Six
Creating records	Seven
Connecting records	Eight
Disconnecting records	Nine
Deleting records	Ten
Schema Access Commands	Eleven
Utilities	Twelve
Boolean operations	Thirteen
Special operators	Fourteen
Multiuser locking	Fifteen
Recovery operators	Sixteen

Chapter seventeen discusses the command statuses that can result from performing DMS commands. The command statuses appear in increasing numerical order. Each command status includes a description of the condition along with its possible causes.

THE ROLE OF DML IN APPLICATION DEVELOPMENT

The MDBS Data Manipulation Language consists of a group of commands, each of which performs some data manipulation task. A DML command is stated within an application program. The programming language used to write an application program is called the host language. A DML command is typically invoked as a function in the host language. When a DML command is given, the MDBS.DMS software carries out the actual data manipulation. This could involve storing data from a host program variable into the data base. Conversely, it could involve extracting data from the data base and depositing it in a host language variable. Many other kinds of data manipulation are supported. In effect, the DML extends the data handling capability of a programming language, so that programs can utilize MDBS data bases in addition to traditional files.

Usage of the MDBS DML in developing an application program depends only upon a knowledge of the data base's logical structure. The application programmer need not be concerned with using pointers, searching indices, disk I/O, file handling, free space management, etc. All of these factors are automatically handled by the MDBS.DMS software. Furthermore, all data manipulation occurs subject to the security and integrity constraints defined in a data base's DDL specification. In a multiuser environment, MDBS.DMS manages record lockout to prevent such problems as one user attempting to modify a record that another user is reading. Special DML commands are available to the application programmer who (beyond the standard passive lockouts) desires to actively lock out records, record types, or sets.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Overview

A data manipulation language provides a procedural approach to data manipulation. It can therefore be contrasted with a query language, which provides a nonprocedural approach to data manipulation. From the standpoint of achieving the highest performance for application software, using a DML is inevitably superior to using a nonprocedural query language. On the other hand, the application developer who is not experienced in using DML can very likely achieve more rapid development by using a nonprocedural query language (at the expense of a sacrifice in performance). In data base management systems that do not support both a DML and a query language, the application developer has no choice. Both kinds of languages are available with MDBS. It is recommended that the query language (see the **MDBS QRS Manual**) be used in situations where the need for rapid development outweighs the need for optimal performance. If optimal performance is the overriding consideration, then the MDBS DML is most appropriate. In addition to QRS, there are other optional access modules including IDML, RDL and BLF.

Although the MDBS DML makes many commands available, it is typically the case that an application developer will need to use only a few kinds of DML commands in any given application program. Across all application programs, there is a small group of about a dozen DML commands that tend to be heavily used. Varying degrees of selective MDBS.DMS linking are permitted, depending on the host language and operating system used (see MDBS System Specific Manuals for details). In other words, only that part of MDBS.DMS that is needed to execute an application program's DML commands is held in main memory. Since the entire MDBS.DMS software is not in main memory, there is more memory available for a larger application program or a larger page buffer region.

One component of MDBS.DMS performs virtual buffer paging, based on an enhanced least-recently-used page replacement algorithm. When a DML command is executed, MDBS.DMS determines whether images of all needed pages are in main memory. If needed pages are not resident, MDBS.DMS has them read into main memory. The larger the page buffer region in main memory, the more likely it is that needed pages are resident and the faster the processing will be. Thus it is important for the application developer to allow as large a page buffer region as possible. In a single-user environment, the page buffer region size is directly assigned by the programmer. In a multiuser environment, MDBS.DMS determines the size of the page buffer region that is shared by all users (i.e., simultaneously executing application programs).

© Copyright 1987 Micro Data Base Systems, Inc.

II. GENERAL BACKGROUND

DATA TRANSFER

Some DML commands pass data from a host language variable into the data base. Others retrieve data from the data base into a host language variable, from which the data can then be used for computation or program output. There are two basic approaches for allowing MDBS.DMS to accomplish the data transfer between a data base and an application program. The approach used depends on the nature of the host programming language. Non-record-oriented languages (such as BASIC) require the data block approach. Host languages with a facility for defining program record types (e.g., COBOL, C, PASCAL) use the program record type approach. Full details of the approach used for a particular host language appear in the MDBS System Specific Manual for that host language.

Data Blocks

A data block is a named sequence of one or more host language variables. MDBS DML commands exist to define, extend, and undefine data blocks (see Chapter XIII). A given host language variable can participate in many data blocks. A DML command to put data into a data base has a data block name as one of its arguments. The value(s) of the host language variable(s) for that data block is(are) put into the data base. A DML command to retrieve data from a data base has a data block name as one of its arguments. The retrieved value(s) becomes the new value(s) for the data block's host language variable(s).

Program Record Types

A program record type plays the same role as a data block. It is a named sequence of one or more host language variables. It is not defined with DML commands, but is specified with the host language's facility for defining program record types (for example, a 01 level entry in a COBOL working storage section, or a record structure of C or PASCAL). A given host language variable usually cannot participate in more than one program record type. A DML command to put data into a data base has a program record type name as one of its arguments. The value(s) of the host language variable(s) for that program record type is(are) put into the data base. A DML command to retrieve data from a data base has a program record type name as one of its arguments. The retrieved value(s) becomes the new value(s) of the program record type's host language variable(s).

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
General Background

CURRENCY INDICATORS

Currency indicators are used by an application programmer to keep track of which record occurrences in a data base are currently of interest. MDBS.DMS maintains two currency indicators for each set specified in a data base's schema: the current owner of the set and the current member of the set. There can be many owner record occurrences in the data base for a given set. At any moment during the execution of an application program no more than one of those owner record occurrences is the current owner of the set. DML commands allow the application programmer to control which (if any) owner record occurrence is the current owner of a set. The SYSTEM record occurrence is always the current owner of every system-owned set (unless it is made null by the programmer). Similarly, there can be many member record occurrences in the data base for a given set. At any moment during the execution of an application program, no more than one of those member record occurrences is the set's current member. DML commands allow the application programmer to control which (if any) member record occurrence is the current member of a set.

As soon as a record occurrence becomes the current owner of a set, you can use the DML to perform these actions:

- find any of its related members through that set (Chapter III)
- retrieve its data values (Chapter IV)
- modify its data values (Chapter V)
- delete the record occurrence from the data base (Chapter X)
- retrieve its record type (Chapter XII)

As soon as a record becomes the current member of a set, you can use the DML to perform these actions:

- find any of its related owners through that set (Chapter III)
- retrieve its data values (Chapter IV)
- modify its data values (Chapter V)
- delete the record occurrence from the data base (Chapter X)
- retrieve its record type (Chapter XII)

Another kind of currency indicator that can be used during data manipulation is called the current of run unit. A run unit is a currently executing application program. There are many record occurrences in a data base. At any moment during the execution of an application program, no more than one record occurrence in the entire data base is the current record of that run unit. The current of a run unit is typically the record that was most recently found by the application program (i.e., by the run unit). DML commands allow the application programmer to control which (if any) data base record is the current of run unit. As soon as a record becomes the current of run unit, you can use the DML to perform these actions:

- delete the record from the data base (Chapter IX)
- modify its data values (Chapter V)
- retrieve its data values (Chapter IV).

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
General Background

Suppose a data base schema has five sets. These can be any mixture of l:l, l:N, N:l, N:M sets. An application program can make use of any of eleven currency indicators. There are five current owner indicators (one per set), five current member indicators (one per set), and the current of run unit. In a multiuser environment, each application program that is using this data base will have its own eleven currency indicators.

In addition to the three kinds of currency indicators already mentioned, MDBS allows an application programmer to optionally define additional currency indicators within the scope of an application program. A maximum of 255 additional user-defined currency indicators can be used within an application program. These additional currency indicators can be usefully employed to "remember" the current owner of some set, before processing causes that current owner to change. The same principle can be applied to current members and the current of run unit. Various DML commands can be used in subsequent processing to access the remembered record. DML commands are also provided for the user (i.e., application programmer) to define additional currency indicators and to make the record that is a set's current owner (or member or current of run unit) the current record for a user-defined currency indicator.

All currency indicators are null when an application program begins execution except the current owner of each system-owned set and the current of run unit, which also has the SYSTEM record as its current record. A currency indicator remains null until a DML command is executed to change it.

STATING A DML COMMAND

Each DML command consists of a short mnemonic. For instance, the mnemonic FFM is used for finding the first member record connected to the current owner of a set. A list of zero, one, two, or three arguments is stated along with a DML command. An argument is either a set name, record type name, area name, data item name, or data block/program record name. For example, FFM needs a single argument: a set name, to indicate for which set we want to find the first member. Correct usage of a DML command depends on a knowledge of the kinds of arguments required by that command. If one of the arguments is a data block (or program record), the application programmer must be aware of whether the command will use the data block's variables as inputs or outputs.

A DML command can use currency indicators and can change currency indicators. An application programmer must know which currency indicators a DML command can use and which currency indicators it can alter (and the nature of the alterations).

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
General Background

Command Status

The execution of a DML command always results in a command status. A command status is an integer in the range from 0 through 255. The command status number is returned as the value of a host language variable of the programmer's choosing. A command status of 0 means that the execution of a command was successfully completed. A command status of 255 means that the data manipulation system tried to find a record occurrence indicated by the command, but that the indicated record occurrence does not exist in the data base. Any other nonzero command status means that the command did not execute normally. Generally speaking, this is caused by either faulty logic of the programmer, a flaw in the command statement, an attempt to violate security or integrity constraints specified in the DDL, failure to have a needed disk on-line, or a hardware malfunction. A detailed description of each command status appears in Chapter XVII.

Command Form

The precise syntax employed for stating a DML command depends on the host language being used. The MDBS System Specific Manual for a given host language shows the syntax for using DML within that host language. There are four major categories of host language interfaces:

- Data block oriented with direct DML invocation (some FORTRANs, some BASICs)
- Data block oriented with indirect DML invocation (some BASICs)
- Record oriented with direct DML invocation (C, some PASCALs)
- Record oriented with indirect DML invocation (some PASCALs, some COBOLs)

A host language is data block oriented if it does not allow program record types; otherwise, it is said to be record oriented. Invoking a DML command directly means that the host language allows a DML mnemonic to be treated as a host language function. This is not allowed in the indirect case, where a DML mnemonic and its associated arguments are parameters of a function. For instance, in the direct case the FFM command has the following generic form:

```
E0 = FFM ("arguments")
```

E0 is a host language variable that receives the command status value returned by the FFM function. In the indirect case, the FFM command has the following generic form:

```
E0 = DMS ("FFM,arguments")
```

E0 is a host language variable that receives the command status returned by the DMS function as a result of executing the FFM command.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
General Background

Command String

When a DML command is stated, the portion of that command within quotes is called the command string. Two elements in a command string are separated by:

- one or more blanks
- a comma
- a comma embedded in a series of blanks

Two consecutive commas, without an intervening nonblank character, indicate a missing element in a command string (such as, "X,,Y"). An incomplete command string (a command string with one fewer than the permitted number of elements) is typically treated as if the rightmost element is missing. That is, if the command string "X,Y" is specified for a command that expects three elements, the first element is "X", the second is "Y", and the rightmost element is missing.

The exception is when the omitted element should have been a data block name (or program record type name) immediately following a data item or record type name (e.g., with the CRS command). In this case, the data block name is assumed to be the same as the record type or data item name (e.g., if Z is a data item name, then "X,Z" is equivalent to "X,Z,Z").

All elements are missing from a command string if there are no elements in the quotes. For some DML commands, it is permissible for an argument in a command string or the entire command string to be missing (such as Find First Sequential, FFS). Some DML commands have no command string (such as, Data Base Close, DBCLS).

Command Descriptions

As each DML command is described in the following chapters, examples of the four different categories of command usage are provided. The mnemonic, arguments, currency indicators used, currency indicators changed, and impact on the data base for each DML command are all independent of the host language used. All of these factors are described in subsequent chapters of the MDBS DMS Manual. Thus the logic employed in using the data manipulation language is the same, regardless of the host languages used.

The syntax for using a command may differ slightly from one host language to another. Consult the appropriate system specific manual for the command syntax required by a given host language.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
General Background

In presenting the DML commands in subsequent Chapters, the following conventions are used:

E0	host language variable for the command status
itm	the name of a data item defined with the DDL
rec	the name of a record type defined with the DDL
set-1, set-2, set-3	the names of sets defined with the DDL
area	the name of an area defined with the DDL
blk	the name of a data block (or program record type)
iblk	the name of a data block (or program record type) whose host language variables provide input to a DML command
oblk	the name of a data block (or program record type) whose host language variables receive output from a DML command
CRU	current of run unit indicator
CO(set)	current owner of the indicated set
CM(set)	current member of the indicated set
CU(i)	current user-defined indicator i ($1 \leq i \leq 255$) for example, CU(7) denotes the current record for the seventh user-defined indicator

Although the description for a DML command may refer to a data block, it should be understood that a program record type takes the place of a data block for record-oriented host languages.

MULTIUSER ENVIRONMENT

In a multiuser environment, many run units can simultaneously share the same copy of serially re-entrant MDBS.DMS code. They also share a common page buffer area. Each run unit has its own program region, which contains the application program and program buffers. A run unit's program buffers are used by MDBS.DMS to manage the specific operations pertinent to that run unit. Each run unit has its own group of currency indicators, which are maintained independently of other run units' currency indicators.

A run unit can lock various record occurrences to prevent them from being altered and/or read by other run units. The rules governing multiuser locks (passive and active) are presented in Chapter XV. Chapter XV includes the DML commands to establish and release locks with the data base control system. It also discusses the DML commands to control the number of attempts (and the time interval length between each attempt) that a DML command will perform to access a locked record.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
General Background

Built into the multiuser MDBS.DMS is a mechanism that allows an application developer to prevent deadlock situations (involving two or more run units). If a run unit attempts an access that will create a deadlock situation, a nonzero command status to that effect is returned for the DML command that attempted the access. The run unit can use this information in its control structure to avoid a deadlock situation.

PROTECTING DATA BASE CONSISTENCY AGAINST EXTERNAL FACTORS

When a run unit opens a data base for processing, it can specify read-only processing. If this is the case, then page images in the page buffer region are always identical to their corresponding pages in the data base (that is, on disk). Thus data base consistency is not damaged by an abnormal interruption (such as a power failure) of the run unit, which does not allow the run unit to close the data base.

If a run unit does not open the data base for read-only processing, then the run unit can alter the data base. Since these alterations are necessarily buffered through the page buffer region, it can happen that a page image in this region is not identical to the corresponding data base page. An abnormal interruption of the run unit could leave the data base inconsistent since the latest changes to page images in main memory might not have been written to the data base. When this occurs, subsequent efforts to open the data base yield the command status 15. A backup copy of the data base must be used in this event or the recovery utility furnished with the RTL form of MDBS can be used to restore the data base. There are two ways to drastically reduce the likelihood of such a situation.

Buffer Flushing

One method makes use of a DML command (DBSAVE) that rewrites all changed page images in MDBS.DMS to the data base. This buffer flushing command can be used wherever and whenever desired in an application program. If, when a run unit abnormally terminates, no changes have been made to the page images in memory since the last flush, then the data base is consistent and error 15 will not occur in a subsequent attempt to open the data base for processing.

Page Image File

The second method is available with the RTL form of MDBS in single user situations. It allows the run unit to specify a page-image file via the PIFD command. This method of preserving consistency is particularly useful if the run unit carries out a complex kind of transaction that can change records on several pages. At some moment during the execution of a complex transaction, some changed page image may have been rewritten into the data base, while other altered images may still reside in the buffer region. An abnormal termination at this point would leave the data base inconsistent. If the data base could be restored to its former state (before the the complex transaction started), the data base would be consistent and the interrupted transaction could be re-initiated. This is the purpose of the page-image file.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
General Background

To declare a complex transaction to use with the page image file, use the TRBGN and TRCOM commands. At any juncture in a run unit, the TRBGN command declares the beginning of a complex transaction. When a commit (TRCOM) command is invoked, the entire complex transaction ends and is committed to the data base. All changes made since TRBGN was last invoked are incorporated into the data base. If an abort (TRABT) command is invoked instead of TRCOM, the complex transaction aborts. No changes that were made since TRBGN was last invoked are incorporated into the data base. If the run unit abnormally terminates after a TRBGN and before a TRABT or TRCOM, then a subsequent command to open the data base finds the data base in the state that existed at the beginning of the complex transaction. With operating systems that do not dynamically allocate file sizes, it is important to create a page image file of sufficient size. Because page-image posting is not meaningful in multiuser situations, it is available only for the single user version of MDBS. The PIFD, TRBGN, TRCOM and TRABT commands are described in Chapter XVI and in the MDBS RTL Manual.

Selective Rollback

The previous mechanisms do not help in the event of a hardware failure that physically destroys a portion of the data base. In this case the RTL transaction logging and recovery facilities are used to restore the data base. The DML commands for utilizing automatic transaction logging within a run unit are presented in Chapter XVI. The transaction log file (defined in the DDL specification) is distinct from the page-image file. Either, both, or neither can be used by a run unit. In the event of <u>any</u> type of inconsistency in the data base (such as abnormal termination, hardware failure, or incorrect data entry), the transaction log file can automatically be applied to a data base back-up copy by using the RTL recovery utility (RCV). This recovery can be selective if desired, so that only certain logged transactions are used to recover. Bad transactions (that is, incorrect data entry) can be discarded during the selective rollback.

Full details about the MDBS recovery facilities available with the RTL form of MDBS, including its use for surveillance of users' activities, are presented in the <u>MDBS Recovery and Transaction Logging Reference Manual</u>.

© Copyright 1987 Micro Data Base Systems, Inc.

III. FIND COMMANDS

The MDBS Data Manipulation Language supports twenty-four find commands, providing an application programmer with many ways to find a given record. The principal objective of any find command is to make a record the current owner of a set, the current member of a set, the current of run unit, or some combination of these. The found record becomes one of the records that the application program is currently interested in; when a record is current, it can be processed in various ways by other DML commands. The **find** commands fall into four main groups:

- Find a set member - given a particular set and a particular owner record of the set, locate a record that is a set member. The find member commands locate the following member record occurrences:

  - first member
  - next member
  - last member
  - prior member
  - member with a particular sort key value
  - next member having the same sort key value as the current set member
  - member with a particular value for some data item
  - next member having an identical data item value as the current set member

  The set member order (specified in the DDL source text) determines which member is first, next, last, or prior. If the member order is sorted, members can be found quickly on the basis of their sort key values.

- Find a set owner - given a particular set and a particular member record of the set, locate a record that is a set owner. The find owner commands locate the following owner record occurrences:

  - first owner
  - next owner
  - last owner
  - prior owner
  - owner with a particular sort key value
  - next owner having the same sort key value as the current set owner
  - owner with a particular value for some data item
  - next owner having an identical data item value as the current set owner

  The set owner order (specified in the DDL source text) determines which owner is first, next, last, or prior. If the owner order is sorted, owners can be found quickly on the basis of their sort key values.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

- Find a calc record - given a record type with a CALC key defined in the DDL source text, any of its record occurrences can be found directly, without using any set in which the record type participates. MDBS.DMS finds the desired record on the basis of a calc key value furnished by the application program. If duplicate calc keys are allowed, then all records with the same calc key value can be found.

- Find a record sequentially - an application program can sequentially find every record occurrence in a desired data base area or areas. This refers to the physical record placement within the data base.

If a find command returns a command status of zero, then the desired record was found. If the command status is 255, then the record requested through a find command does not exist in the data base. The branching and control structure of a host language program is typically based quite heavily on the command status returned by the various find commands used in the program.

Data security conditions, as defined in a DDL schema specification, are automatically enforced when a find command is invoked. If the user of an application program attempts to perform a "find" that is inconsistent with the user's read access codes, a nonzero command status results that indicates the find was not performed.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

COMMAND DETAILS

FDRK **Find Duplicate Record based on calc Key** **FDRK**

Command Arguments **FDRK,rec,iblk**

Currency Indicators

Used	Changed	Change Description
CRU	CRU	record with duplicate calc key value

Description

If the current of run unit is not null when FDRK is invoked:
Find an occurrence of the same record type as the current of run unit, and having a .i.calc key; value that matches the calc key value held in **iblk**'s variable(s). If such a record is found, it becomes the new current of run unit. This command is valid only if the record that is the current of run unit is an occurrence of **rec** and **rec** has been defined (in the DDL) as having a calc key. The calc key value indicated by **iblk** must duplicate the calc key value of the current of run unit; otherwise, a nonzero command status results (the current of run unit is not altered).

Use of FRK, followed by repeated use of FDRK, will find all records of a given record type having the same calc key value. The order in which these are found is not guaranteed. If an FDRK command is given when there are no more duplicates of a particular calc key value, then the current of run unit becomes null and the command status is 255. If a user does not have read access to all data items that form the calc key, then a nonzero command status is returned.

If the current of run unit is null when FDRK is invoked:
FDRK has exactly the same effect as FRK.

Examples of Command Usage

```
block/direct   .. E0 = FDRK ("rec,iblk")
block/indirect .. E0 = DMS ("FDRK,rec,iblk")
record/direct  .. E0 = FDRK ("rec", iblk)
record/indirect.. E0 = DMSD ("FDRK,rec", iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FFM **Find First Member** **FFM**

Command Arguments **FFM,set–1**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	first member (set-1)
	CRU	first member (set-1)

Description

The first member connected to the current owner of **set-1** becomes the .i.current member; of **set-1** and the current of the run unit. Which member is first depends on the **set-1** member order, as specified with the DDL. If the current owner of **set-1** has no member, then the command status is 255, the current of run unit becomes null, and the current member of **set-1** is null. If a user does not have read access to **set-1**, then a nonzero command status is returned.

Examples of Command Usage

```
block/direct   .. E0 = FFM ("set-1")
block/indirect .. E0 = DMS ("FFM,set-1")
record/direct  .. E0 = FFM ("set-1")
record/indirect.. E0 = DMS ("FFM,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FFO **Find First Owner** **FFO**

Command Arguments **FFO,set–1**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	first owner (set-1)
	CRU	first owner (set-1)

Description

The first owner connected to the current member of **set-1** becomes the .i.current owner; of **set-1** and the current of the run unit. Which owner is first depends on the **set-1** owner order, as specified with the DDL. If the current member of **set-1** has no owner, then the command status is 255, the current of run unit is null, and the current owner of **set-1** is null. If a user does not have read access to **set-1**, then a nonzero command status is returned.

Examples of Command Usage

```
block/direct   .. E0 = FFO ("set-1")
block/indirect .. E0 = DMS ("FFO,set-1")
record/direct  .. E0 = FFO ("set-1")
record/indirect.. E0 = DMS ("FFO,set-1")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FFS

Find First Sequential record

FFS

Command Arguments **FFS,area (the area argument is optional)**

Currency Indicators

Used	Changed	Change Description
none	CRU	first record in area

Description

The record that is physically first within the indicated **area** becomes the current of run unit. If the indicated **area** has no records, then current of run unit becomes null and the command status is 255. When the **area** argument is omitted, FFS operates on the main **area** and then on all subsequent **areas** until a record is found (i.e., the main area may have no data records).

FFS allows a user with any access code to find the physically first record of an **area** regardless of the access codes of that record's record type. However, the user will be unable to retrieve data from the found record or modify its data unless the user's access authorization permits read or write access to occurrences of the record type.

Examples of Command Usage

```
block/direct    .. E0 = FFS ("area")      or E0 = FFS ("    ")
block/indirect .. E0 = DMS ("FFS,area") or E0 = DMS ("FFS")
record/direct   .. E0 = FFS ("area")      or E0 = FFS ("    ")
record/indirect.. E0 = DMS ("FFS,area") or E0 = DMS ("FFS")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FLM — Find Last Member

FLM

Command Arguments FLM,set-1

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	last member (set-1)
	CRU	last member (set-1)

Description

The last member connected to the current owner of **set-1** becomes the current member of **set-1** and the current of the run unit. Which member is last depends on the **set-1** member order, as specified with the DDL. If the current owner of **set-1** has no member, then the command status is 255, the current of run unit becomes null and the current member of **set-1** becomes null. If a user does not have read access to **set-1**, then a nonzero command status is returned.

Examples of Command Usage

```
block/direct   .. E0 = FLM ("set-1")
block/indirect .. E0 = DMS ("FLM,set-1")
record/direct  .. E0 = FLM ("set-1")
record/indirect.. E0 = DMS ("FLM,set-1")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FLO **Find Last Owner** **FLO**

Command Arguments **FLO,set–1**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	last owner (set-1)
	CRU	last owner (set-1)

Description

The last owner connected to the current member of **set-1** becomes the current owner of **set-1** and the current of the run unit. Which owner is last depends on the **set-1** owner order, as specified with the DDL. If the current member of **set-1** has no owner, then the command status is 255, the current of run unit becomes null, and the current owner of **set-1** becomes null. If a user does not have read access to **set-1**, then a nonzero command status is returned.

Examples of Command Usage

```
block/direct   .. E0 = FLO ("set-1")
block/indirect .. E0 = DMS ("FLO,set-1")
record/direct  .. E0 = FLO ("set-1")
record/indirect.. E0 = DMS ("FLO,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FMI **Find Member based on data Item** **FMI**

Command Arguments **FMI,item,set−1,iblk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	first member (set-1) with data item value
	CRU	first member (set-1) with data item value

Description

The first member record connected to the current owner of **set-1** which has a value for **itm** that matches the value of **iblk** becomes the current member of **set-1**. It also becomes the current of run unit. If there are two or more member records for the current owner of **set-1**, all having the **iblk** value for **itm**, which one is first depends on the **set-1** member order. The command FNMI can be used to find the next member having the same **iblk** value for **itm**.

The host language variable of **iblk** must be consistent in size and type with the **itm** data item (as specified in the DDL). If the value of **iblk**'s host language variable does not match the **itm** value of any member record connected to the current owner of **set- 1**, then the command status is 255, the current member of **set-1** becomes null, and the current of run unit becomes null. A nonzero command status is returned if a user does not have read access to both **itm** and **set-1**.

Examples of Command Usage

```
block/direct   .. E0 = FMI ("itm,set-1,iblk")
block/indirect .. E0 = DMS ("FMI,itm,set-1,iblk")
record/direct  .. E0 = FMI ("itm,set-1",iblk)
record/indirect.. E0 = DMSD ("FMI,itm,set-1",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FMSK **Find Member based on Sort Key** **FMSK**

Command Arguments **FMSK,set-1,iblk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	first member (set-1) with sort key value
CRU		first member (set-1) with sort key value

Description

The first member record that has the sort key value of **iblk** and that is connected to the current owner of **set-1** becomes the current member of **set-1**. It also becomes the current of the run unit. **Set-1** must have a sorted member order.

If several member records for the current owner of **set-1** have the same sort key value, the one that is first depends on the DDL sort-clause specification for the **set-1** member order (either LIFO, FIFO, or IMMATERIAL). The host language variable(s) of **iblk** must contain the desired sort key value. If there are multiple data items for the **set-1** sort key, the sort key value for **iblk** must be consistent with the sequence, types and sizes of the data items in the DDL sort key specification. If RECORD-TYPE is part of the sort key, then the name of the desired record type must appear as part of the **iblk** sort key value; the host language variable holding the record type name must be consistent with a string data item of 8 bytes in size.

If the current owner of **set-1** has no member whose sort key value matches the sort key value of **iblk**, then the command status is 255. The member whose sort key value immediately follows the **iblk** sort key value becomes the current member of **set-1**, as well as the current of run unit. The exceptions are when the current owner of **set-1** has no members or when **iblk**'s sort key value is greater than the sort key value of the last member of **set-1**'s current owner. In these cases, the current member of **set-1** becomes null and the current of run unit becomes null.

A nonzero command status is returned if a user does not have read access to **set-1** and all data items that make up the member sort key.

Examples of Command Usage

```
block/direct   .. E0 = FMSK ("set-1,iblk")
block/indirect .. E0 = DMS ("FMSK,set-1,iblk")
record/direct  .. E0 = FMSK ("set-1",iblk)
record/indirect.. E0 = DMSD ("FMSK,set-1",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FNM **Find Next Member** **FNM**

Command Arguments **FNM,set-1**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	next member (set-1)
CM(set-1)	CRU	next member (set-1)

Description

If the current member of **set-1** is not null when FNM is invoked:
The next member record (connected to the current owner of **set-1**) following the current member of **set-1** becomes the new current member of **set-1**. It also becomes the current of run unit. Which member record is logically next depends on the **set-1** member order, as specified with the DDL. If the current member of **set-1** is the last member for the current owner of **set-1**, then FNM cannot find a next member. When there is no next member for the current owner of **set-1**, the command status is 255, the current member of **set-1** becomes null, and the current of run unit becomes null. A nonzero command status is returned if a user does not have read access to **set-1**.

If the current member of **set-1** is null when FNM is invoked:
FNM has exactly the same effect as FFM.

Examples of Command Usage

```
block/direct   .. E0 = FNM ("set-1")
block/indirect .. E0 = DMS ("FNM,set-1")
record/direct  .. E0 = FNM ("set-1")
record/indirect.. E0 = DMS ("FNM,set-1")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FNMI — Find Next Member based on data Item

FNMI

Command Arguments

FNMI,itm,set−1,iblk

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	next member (set-1) with item value
CM(set-1)	CRU	next member (set-1) with item value

Description

If the current member of **set-1** is not null when FNMI is invoked:
The next member record connected to the current owner of **set-1** which has a value for **itm** that matches the value of **iblk**, becomes the current member of **set-1**. It also becomes the current of run unit. The host language variable of **iblk** must be consistent in size and type with the **itm** data item (as specified in the DDL). If there are two or more member records following the current member of **set-1**, all having the **iblk** value for **itm**, which one is next depends on the **set-1** member order. If such a record does not exist, the command status is 255, the current member of **set-1** becomes null, and the current of run unit becomes null. If a user does not have read access to both **itm** and **set-1**, then a nonzero command status is returned.

If the current member of **set-1** is null when FNMI is invoked:
FNMI has exactly the same effect as FMI. The first member record connected to the current owner of **set-1**, and having a value for **itm** that matches the value of **iblk**, becomes the current member of **set-1**.

Examples of Command Usage

```
block/direct   .. E0 = FNMI ("itm,set-1,iblk")
block/indirect .. E0 = DMS ("FNMI,itm,set-1,iblk")
record/direct  .. E0 = FNMI ("itm,set-1",iblk)
record/indirect.. E0 = DMSD ("FNMI,itm,set-1",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FNMSK — Find Next Member based on Sort Key — FNMSK

Command Arguments

FNMSK,set−1,iblk

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	next member (set-1) with sort key value
CM(set-1)	CRU	next member (set-1) with sort key value

Description

The next member record (after the current member) that has the sort key value of **iblk** and that is connected to the current owner of **set-1** becomes the current member of **set-1**. It also becomes the current of the run unit. **Set-1** must have a sorted member order.

If **set-1** has no current member, then FNMSK behaves just as FMSK. If several member records for the current owner of **set-1** have the same sort key value, the one that is next depends on the DDL sort-clause specification for the **set-1** member order (either LIFO, FIFO, or IMMATERIAL). The host language variable(s) of **iblk** must contain the desired sort key value. If there are multiple data items for the **set-1** sort key, the sort key value for **iblk** must be consistent with the sequence, types and sizes of the data items in the DDL sort key specification. If RECORD-TYPE is part of the sort key, then the name of the desired record type must appear as part of the **iblk** sort key value; the host language variable holding the record type name must be consistent with a string data item of 8 bytes in size.

If the owner of **set-1** has no member postioned between the current member and the end of the set with a sort key value that matches the value of **iblk**, the command status becomes 255. The next member immediately following the current member becomes the current member of **set-1**, as well as the current of run unit. The exceptions are when the current owner of **set-1** has no members or when **iblk**'s sort key value is greater than the sort key value of the last member of **set-1**'s current owner. In these cases, the current member of **set-1** becomes null and the current of run unit becomes null.

A nonzero command status is returned if a user does not have read access to **set-1** and all data items that make up the member sort key.

Examples of Command Usage

```
block/direct   .. E0 = FNMSK ("set-1,iblk")
block/indirect .. E0 = DMS ("FNMSK,set-1,iblk")
record/direct  .. E0 = FNMSK ("set-1",iblk)
record/indirect.. E0 = DMSD ("FNMSK,set-1",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FNO **Find Next Owner** **FNO**

Command Arguments **FNO,set-1**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	next owner (set-1)
CO(set-1)	CRU	next owner (set-1)

Description

If the current owner of **set-1** is not null when FNO is invoked:
The next owner record (connected to the current member of **set-1**) following the current owner of **set-1** becomes the new current owner of **set-1**. It also becomes the current of run unit. Which owner record is logically next depends on the **set-1** owner order, as specified with the DDL. If the current owner of **set-1** is the last owner for the current member of **set-1**, then FNO cannot find a next owner. When there is no next owner for the current member of **set-1**, the command status is 255, the current owner of **set-1** becomes null, and the current of run unit becomes null. A nonzero command status is returned if a user does not have read access to **set-1**.

If the current owner of **set-1** is null when FNO is invoked:
FNO has exactly the same effect as FFO.

Examples of Command Usage

```
block/direct   .. E0 = FNO ("set-1")
block/indirect .. E0 = DMS ("FNO,set-1")
record/direct  .. E0 = FNO ("set-1")
record/indirect.. E0 = DMS ("FNO,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FNOI — Find Next Owner based on data Item

FNOI

Command Arguments FNOI,itm,set−1,iblk

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	next owner(set-1) with item value
CO(set-1)	CRU	next owner(set-1) with item value

Description

If the current owner of **set-1** is not null when FNOI is invoked:
The next owner record connected to the current member of **set-1** which has a value for **itm** that matches the value of **iblk** becomes the current owner of **set-1**. It also becomes the current of run unit. The host language variable of **iblk** must be consistent in size and type with the **itm** data item (as specified in the DDL). If there are two or more owner records following the current owner of **set-1**, all having the **iblk** value for **itm**, which one is next depends on the **set-1** owner order. If such a record does not exist, the command status is 255, the current owner of **set-1** becomes null, and the current of run unit becomes null. If a user does not have read access to both **itm** and **set-1**, then a nonzero command status is returned.

If the current owner of **set-1** is null when FNOI is invoked:
FNOI has exactly the same effect as FOI. The first owner record connected to the current member of **set-1**, and having a value for **itm** that matches the value of **iblk**, becomes the current owner of **set-1**.

Examples of Command Usage

```
block/direct   .. E0 = FNOI ("itm,set-1,iblk")
block/indirect .. E0 = DMS ("FNOI,itm,set-1,iblk")
record/direct  .. E0 = FNOI ("itm,set-1",iblk)
record/indirect.. E0 = DMSD ("FNOI,itm,set-1",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FNOSK — Find Next Owner based on Sort Key — FNOSK

Command Arguments **FNOSK,set-1,iblk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CO(set-1)	next owner (set-1) with sort key value
CM(set-1)	CRU	next owner (set-1) with sort key value

Description

The next owner record (after the current owner) that has the sort key value of **iblk** and that is connected to the current member of **set-1** becomes the current owner of **set-1**. It also becomes the current of the run unit. **Set-1** must have a sorted owner order.

If **set-1** has no current owner, then FNOSK behaves just as FOSK. If several owner records for the current member of **set-1** have the same sort key value, the one that is next depends on the DDL sort-clause specification for the **set-1** owner order (either LIFO, FIFO, or IMMATERIAL). The host language variable(s) of **iblk** must contain the desired sort key value. If there are multiple data items for the **set-1** sort key, the sort key value for **iblk** must be consistent with the sequence, types and sizes of the data items in the DDL sort key specification. If RECORD-TYPE is part of the sort key, then the name of the desired record type must appear as part of the **iblk** sort key value; the host language variable holding the record type name must be consistent with a string data item of 8 bytes in size.

If the current member of **set-1** has no owner after the current owner whose sort key value matches the sort key value of **iblk**, then the command status is 255. The next owner immediately following the current member becomes the current owner of **set-1**, as well as the current of run unit. The exceptions are when the current member of **set-1** has no owners or when **iblk**'s sort key value is greater than the sort key value of the last owner of **set-1**'s current member. In these cases, the current owner of **set-1** becomes null and the current of run unit becomes null.

A nonzero command status is returned if a user does not have read access to **set-1** and all data items that make up the owner sort key.

Examples of Command Usage

```
block/direct   .. E0 = FNOSK ("set-1,iblk")
block/indirect .. E0 = DMS ("FNOSK,set-1,iblk")
record/direct  .. E0 = FNOSK ("set-1",iblk)
record/indirect.. E0 = DMSD ("FNOSK,set-1",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FNS **Find Next Sequential record** **FNS**

Command Arguments **FNS,area (the area argument is optional)**

Currency Indicators

Used	Changed	Change Description
CRU	CRU	next record in area

Description

The record that is physically next after the current of run unit within the indicated **area** becomes the current of run unit. A nonzero command status is issued if the current of run unit is not in the indicated **area**; the current of run unit is not changed.

If the current of run unit is the physically last record in an **area** when FNS is invoked, then FNS will give the command status of 255 and the current of run unit is unchanged. If an **area** name is not indicated, then FNS will not give a 255 command status at the end of an **area**. It will proceed to make the physically first record of the next **area** the current of run unit. The sequence of areas is determined by the order in which areas are defined in the DDL **area** section. The main **area** is always the first **area**. If the current of run unit is the last record in the last **area**, invoking FNS will give a 255 command status and leave the current of run unit null.

FNS allows a user with any access code to find the physically next record of an **area** regardless of the access codes of that record's record type. However, the user will be unable to retrieve data from the found record or modify its data unless the user's access authorization permits read or write access to occurrences of the record type.

Examples of Command Usage

```
block/direct   .. E0 = FNS ("area")      or E0 = FNS ("   ")
block/indirect .. E0 = DMS ("FNS,area") or E0 = DMS ("FNS")
record/direct  .. E0 = FNS ("area")      or E0 = FNS ("   ")
record/indirect.. E0 = DMS ("FNS,area") or E0 = DMS ("FNS")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FOI — Find Owner based on data Item

FOI

Command Arguments

FOI,item,set−1,iblk

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	first owner (set-1) with data item value
CRU		first owner (set-1) with data item value

Description

The first owner record connected to the current member of **set-1** which has a value for **itm** that matches the value of **iblk** becomes the current owner of **set-1**. It also becomes the current of run unit. If there are two or more owner records for the current member of **set-1**, all having the **iblk** value for **itm**, which one is first depends on the **set-1** owner order. The command FNOI can be used to find the next owner having the same **iblk** value for **itm**.

The host language variable of **iblk** must be consistent in size and type with the **itm** data item (as specified in the DDL). If the value of **iblk**'s host language variable does not match the **itm** value of any owner record connected to the current member of **set- 1**, then the command status is 255, the current owner of **set-1** becomes null, and the current of run unit becomes null. A nonzero command status is returned if a user does not have read access to both **itm** and **set-1**.

Examples of Command Usage

```
block/direct   .. E0 = FOI ("itm,set-1,iblk")
block/indirect .. E0 = DMS ("FOI,itm,set-1,iblk")
record/direct  .. E0 = FOI ("itm,set-1",iblk)
record/indirect.. E0 = DMSD ("FOI,itm,set-1",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FOSK **Find Owner based on Sort Key** **FOSK**

Command Arguments **FOSK,set-1,iblk**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	first owner (set-1) with sort key value
	CRU	first owner (set-1) with sort key value

Description

The first owner record that has the sort key value of **iblk** and that is connected to the current member of **set-1** becomes the current owner of **set-1**. It also becomes the current of the run unit. **Set-1** must have a sorted owner order.

If several owner records for the current member of **set-1** have the same sort key value, the one that is first depends on the DDL sort-clause specification for the **set-1** owner order (either LIFO, FIFO, or IMMATERIAL). The host language variable(s) of **iblk** must contain the desired sort key value. If there are multiple data items for the **set-1** sort key, the sort key value for **iblk** must be consistent with the sequence, types and sizes of the data items in the DDL sort key specification. If RECORD-TYPE is part of the sort key, then the name of the desired record type must appear as part of the **iblk** sort key value; the host language variable holding the record type name must be consistent with a string data item of 8 bytes in size.

If the current member of **set-1** has no owner whose sort key value matches the sort key value of **iblk**, then the command status is 255. The owner whose sort key value immediately follows the **iblk** sort key value becomes the current owner of **set-1**, as well as the current of run unit. The exceptions are when the current member of **set-1** has no owners or when **iblk**'s sort key value is greater than the sort key value of the last owner of **set-1**'s current member. In these cases, the current owner of **set-1** becomes null and the current of run unit becomes null.

A nonzero command status is returned if a user does not have read access to **set-1** and all data items that make up the owner sort key.

Examples of Command Usage

```
block/direct   .. E0 = FOSK ("set-1,iblk")
block/indirect .. E0 = DMS ("FOSK,set-1,iblk")
record/direct  .. E0 = FOSK ("set-1",iblk)
record/indirect.. E0 = DMSD ("FOSK,set-1",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FPM **Find Prior Member** **FPM**

Command Arguments **FPM,set−1**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	prior member(set-1)
CM(set-1)	CRU	prior member(set-1)

Description

If the current member of **set-1** is not null when FPM is invoked:
The member record (connected to the current owner of **set-1**) preceding the current member of **set-1** becomes the new current member of **set-1**. It also becomes the current of run unit. Which member record is logically prior depends on the **set-1** member order, as specified with the DDL. If the current member of **set-1** is the first member for the current owner of **set-1**, then FPM cannot find a prior member. When there is no prior member for the current owner of **set-1**, the command status is 255, the current member of **set-1** becomes null, and the current of run unit becomes null. A nonzero command status is returned if a user does not have read access to **set-1**.

If the current member of **set-1** is null when FPM is invoked:
FPM has exactly the same effect as FLM.

Examples of Command Usage

```
block/direct   .. E0 = FPM ("set-1")
block/indirect .. E0 = DMS ("FPM,set-1")
record/direct  .. E0 = FPM ("set-1")
record/indirect.. E0 = DMS ("FPM,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FPMI — Find Prior Member based on data Item — FPMI

Command Arguments

FPMI,itm,set−1,iblk

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	prior member (set-1) with item value
CM(set-1)	CRU	prior member (set-1) with item value

Description

If the current member of **set-1** is not null when FPMI is invoked:
The member record connected to the current owner of **set-1** that precedes the current member and has a value for **itm** that matches the value of **iblk** becomes the current member of **set-1**. It also becomes the current of run unit. The host language variable of **iblk** must be consistent in size and type with the **itm** data item (as specified in the DDL). If there are two or more member records preceding the current member of **set-1**, all having the **iblk** value for **itm**, which one is found depends on the **set-1** member order. If such a record does not exist, the command status is 255, the current member of **set-1** becomes null, and the current of run unit becomes null. If a user does not have read access to both **itm** and **set-1**, then a nonzero command status is returned.

If the current member of **set-1** is null when FPMI is invoked:
The last member record connected to the current owner of **set-1**, and having a value for **itm** that matches the value of **iblk**, becomes the current member of **set-1**.

Examples of Command Usage

```
block/direct   .. E0 = FPMI ("itm,set-1,iblk")
block/indirect .. E0 = DMS ("FPMI,itm,set-1,iblk")
record/direct  .. E0 = FPMI ("itm,set-1",iblk)
record/indirect.. E0 = DMSD ("FPMI,itm,set-1",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FPMSK **Find Prior Member based on Sort Key** **FPMSK**

Command Arguments **FPMSK,set-1,iblk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	prior member (set-1) with sort key value
CRU		prior member (set-1) with sort key value

Description

Set-1 must have a sorted member order. The member record prior to the current member that has the sort key value of **iblk** and that is connected to the current owner of **set-1** becomes the current member of **set-1**. It also becomes the current of the run unit. If **set-1** has no current member, then the last member having the sort key value is found. If several member records for the current owner of **set-1** have the same sort key value, the one that is found depends on the DDL sort-clause specification for the **set-1** member order (either LIFO, FIFO, or IMMATERIAL).

The host language variable(s) of **iblk** must contain the desired sort key value. If there are multiple data items for the **set-1** sort key, the sort key value for **iblk** must be consistent with the sequence, types and sizes of the data items in the DDL sort key specification. If RECORD-TYPE is part of the sort key, then the name of the desired record type must appear as part of the **iblk** sort key value; the host language variable holding the record type name must be consistent with a string data item of 8 bytes in size.

If the current owner of **set-1** has no member before the current member whose sort key value matches the sort key value of **iblk**, then the command status is 255. The member immediately preceding the current member becomes the current member of **set-1**, as well as the current of run unit. The exceptions are when the current owner of **set-1** has no members or when **iblk**'s sort key value is lower than the sort key value of the first member of **set-1**'s current owner. In these cases, the current member of **set-1** becomes null and the current of run unit becomes null.

A nonzero command status is returned if a user does not have read access to **set-1** and all data items that make up the member sort key.

Examples of Command Usage

```
block/direct   .. E0 = FPMSK ("set-1,iblk")
block/indirect .. E0 = DMS ("FPMSK,set-1,iblk")
record/direct  .. E0 = FPMSK ("set-1",iblk)
record/indirect.. E0 = DMSD ("FPMSK,set-1",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FPO **Find Prior Owner** **FPO**

Command Arguments **FPO,set–1**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	prior owner (set-1)
CO(set-1)	CRU	prior owner (set-1)

Description

If the current owner of **set-1** is not null when FPO is invoked:
The owner record (connected to the current member of **set-1**) preceding the current owner of **set-1** becomes the new current owner of **set-1**. It also becomes the current of run unit. Which owner record is logically prior depends on the **set-1** owner order, as specified with the DDL. If the current owner of **set-1** is the first owner for the current member of **set-1**, then FPO cannot find a prior owner. When there is no prior owner for the current member of **set-1**, the command status is 255, the current owner of **set-1** becomes null, and the current of run unit becomes null. A nonzero command status is returned if a user does not have read access to **set-1**.

If the current owner of **set-1** is null when FPO is invoked:
FPO has exactly the same effect as FLO.

Examples of Command Usage

```
block/direct   .. E0 = FPO ("set-1")
block/indirect .. E0 = DMS ("FPO,set-1")
record/direct  .. E0 = FPO ("set-1")
record/indirect.. E0 = DMS ("FPO,set-1")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FPOI — Find Prior Owner based on data Item

FPOI

Command Arguments

FPOI,itm,set−1,iblk

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	prior owner (set-1) with item value
CO(set-1)	CRU	prior owner (set-1) with item value

Description

If the current owner of **set-1** is not null when FPOI is invoked:
The prior owner record connected to the current member of **set-1** which has a value for **itm** that matches the value of **iblk**, becomes the current owner of **set-1**. It also becomes the current of run unit. The host language variable of **iblk** must be consistent in size and type with the **itm** data item (as specified in the DDL). If there are two or more owner records preceding the current owner of **set-1**, all having the **iblk** value for **itm**, which one is prior depends on the **set-1** owner order. If such a record does not exist, the command status is 255, the current owner of **set-1** becomes null, and the current of run unit becomes null. If a user does not have read access to both **itm** and **set-1**, then a nonzero command status is returned.

If the current owner of **set-1** is null when FPOI is invoked:
The last owner record connected to the current member of **set-1**, and having a value for **itm** that matches the value of **iblk**, becomes the current owner of **set-1**.

Examples of Command Usage

```
block/direct   .. E0 = FPOI ("itm,set-1,iblk")
block/indirect .. E0 = DMS ("FPOI,itm,set-1,iblk")
record/direct  .. E0 = FPOI ("itm,set-1",iblk)
record/indirect.. E0 = DMS ("FPOI,itm,set-1",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FPOSK — Find Prior Owner based on Sort Key — FPOSK

Command Arguments **FPOSK,set-1,iblk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CO(set-1)	prior owner (set-1) with sort key value
CM(set-1)	CRU	prior owner (set-1) with sort key value

Description

Set-1 must have a sorted owner order. The prior owner record (before the current owner) that has the sort key value of **iblk** and that is connected to the current member of **set-1** becomes the current owner of **set-1**. It also becomes the current of the run unit. If **set-1** has no current owner, then the last owner having the sort key value is found. If several owner records for the current member of **set-1** have the same sort key value, the one that is prior depends on the DDL sort-clause specification for the **set-1** owner order (either LIFO, FIFO, or IMMATERIAL).

The host language variable(s) of **iblk** must contain the desired sort key value. If there are multiple data items for the **set-1** sort key, the sort key value for **iblk** must be consistent with the sequence, types and sizes of the data items in the DDL sort key specification. If RECORD-TYPE is part of the sort key, then the name of the desired record type must appear as part of the **iblk** sort key value; the host language variable holding the record type name must be consistent with a string data item of 8 bytes in size.

If the current member of **set-1** has no owner before the current owner whose sort key value matches the sort key value of **iblk**, then the command status is 255. The owner immediately preceding the current owner becomes the current owner of **set-1**, as well as the current of run unit. The exceptions are when the current member of **set-1** has no owners or when **iblk**'s sort key value is less than the sort key value of the first owner of **set- 1**'s current member. In these cases, the current owner of **set-1** becomes null and the current of run unit becomes null.

A nonzero command status is returned if a user does not have read access to **set-1** and all data items that make up the owner sort key.

Examples of Command Usage

```
block/direct   .. E0 = FPOSK ("set-1,iblk")
block/indirect .. E0 = DMS ("FPOSK,set-1,iblk")
record/direct  .. E0 = FPOSK ("set-1",iblk)
record/indirect.. E0 = DMSD ("FPOSK,set-1",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Find Commands

FRK **Find Record based on calc Key** **FRK**

Command Arguments **FRK,rec,iblk**

Currency Indicators

Used	Changed	Change Description
none	CRU	record with calc key value

Description

Find the occurrence of the indicated record type (**rec**) whose calc key has a value that matches the value of **iblk**'s host language variable(s). If such an occurrence is found, it becomes the current of run unit. This command is valid only if **rec** has a calc key clause in the DDL specification. Occurrences of **rec** with duplicate calc key values may exist in the data base if the DDL specification declares that duplicates are allowed. If there is more than one occurrence of **rec** whose calc key value matches the **iblk** value, then FRK finds one of these duplicates and FDRK can be used repeatedly to find each of the other duplicates.

The sequence, size and type of **iblk**'s host language variables must be consistent with the sequence, size and type of data items that make up the calc key for **rec**. If the **iblk** value does not match the calc key value for any occurrence of the **rec** record type, then the command status is 255 and CRU becomes null. A nonzero command status is returned if a user does not have read access to all data items that make up **rec**'s calc key.

Examples of Command Usage

```
block/direct   .. E0 = FRK ("rec,iblk")
block/indirect .. E0 = DMS ("FRK,rec,iblk")
record/direct  .. E0 = FRK ("rec", iblk)
record/indirect.. E0 = DMSD ("FRK,rec", iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

IV. RETRIEVAL COMMANDS

The MDBS Data Manipulation Language supports twenty-eight data retrieval commands. Each retrieval command transfers data from a record to one or more host language variables through a data block or a host language data structure. The retrieval commands form two major groups:

- **get** commands - get data from a **current** record
- **obtain** commands - find a record, make the record current, and then **obtain** data from it

Get Commands

The get commands retrieve data from a record that is the current owner of a specific set, the current member of a specific set, or the current of run unit. The **get** commands can retrieve an entire record occurrence or a single data item value.

In retrieving an entire record occurrence, the get commands transfer the values of all data items (fields) from the record to the appropriate host language variables.

In retrieving a single data item value, the get commands transfer the value of a particular data item (field) from the record to the appropriate host language program variable. This command subgroup provides maximum data independence.

Obtain Commands

The obtain commands combine the actions of the find commands (from the previous chapter) with the actions of the get commands, producing a set of commands that locate and retrieve an entire record occurrence in one step.

Each obtain command retrieves a desired record by first making it the current owner of an indicated set, the current member of an indicated set, or the current of run unit. The data values of the entire record are deposited in the appropriate host language variables. This retrieval approach results in faster, more efficient processing, particularly in multiuser environments.

Host Language Variables

For both retrieval command groups, define each host language variable that receives retrieved data to be consistent with the corresponding data base data item in type and size. MDBS III converts each stored data value from its internal MDBS form into the consistent host language form. If the size of a host language variable is insufficient to hold the entire data value, the data base control system returns a nonzero command status.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

When retrieving data values for an entire record occurrence, match the host language data structure (data block or program record) with the corresponding record type. Specifically, make sure each host language variable of the data structure corresponds in sequence with each data item in the record type (as defined in the DDL source text).

The data base control system automatically enforces data security conditions (given in a DDL schema specification) when a retrieval command is invoked. If the user of an application program attempts to retrieve information that is inconsistent with his read access codes, MDBS.DMS does not retrieve the data values and issues a nonzero command status which indicates the retrieval was not performed.

If the data base control system returns any command status other than zero, the retrieval command does not transfer any data to the host language data structure and the host language variables remain unchanged.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

COMMAND DETAILS

GETC — GET data from Current of run unit — GETC

Command Arguments GETC,oblk

Currency Indicators

Used	Changed	Change Description
CRU	None	

Description

All data values in the record that is the .i.current of run unit; are returned in the host language variables of **oblk.** The types, sizes and sequence of these variables must be consistent with the data items that make up the record type for the current of run unit. A nonzero command status is returned if a user does not have read access to all data items that make up the record type for the current of run unit.

Examples of Command Usage

```
block/direct   .. E0 = GETC ("oblk")
block/indirect .. E0 = DMS ("GETC,oblk")
record/direct  .. E0 = GETC (oblk)
record/indirect.. E0 = DMSD ("GETC",oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

GETM — GET data from Member

GETM

Command Arguments **GETM,set—1,oblk**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	None	

Description

All data values in the .i.current member; of **set-1** are returned in the host language variables of **oblk.** The types, sizes and sequence of these variables must be consistent with the data items that make up the member record type of **set-1.** A nonzero command status is returned if a user does not have read access to all data items of the member record type of **set-1.**

Examples of Command Usage

```
block/direct   .. E0 = GETM ("set-1,oblk")
block/indirect .. E0 = DMS ("GETM,set-1,oblk")
record/direct  .. E0 = GETM ("set-1",oblk)
record/indirect.. E0 = DMSD ("GETM,set-1",oblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

GETO — GET data from Owner

GETO

Command Arguments GETO,set-1,oblk

Currency Indicators

Used	Changed	Change Description
CO(set-1)	None	

Description

All data values in the .i.current owner; of **set-1** are returned in the host language variables of **oblk**. The types, sizes and sequence of these variables must be consistent with the data items that make up the owner record type of **set-1**. A nonzero command status is returned if a user does not have read access to all data items of the owner record type of **set-1**.

Examples of Command Usage

```
block/direct   .. E0 = GETO ("set-1,oblk")
block/indirect .. E0 = DMS ("GETO,set-1,oblk")
record/direct  .. E0 = GETO ("set-1",oblk)
record/indirect.. E0 = DMSD ("GETO,set-1",oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

GFC — Get Field from Current of run unit

GFC

Command Arguments GFC,itm,oblk

Currency Indicators

Used	Changed	Change Description
CRU	None	

Description

The value of the **itm** data item (field) in the record that is the current of run unit is returned in the host language variable of **oblk**. The type and size of this variable must be consistent with the type and size of **itm** in the DDL specification. A nonzero command status is returned if a user does not have read access to the **itm** field.

Examples of Command Usage

```
block/direct   .. E0 = GFC ("itm,oblk")
block/indirect .. E0 = DMS ("GFC,itm,oblk")
record/direct  .. E0 = GFC ("itm",oblk)
record/indirect.. E0 = DMSD ("GFC,itm",oblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

GFM Get Field from Member **GFM**

Command Arguments **GFM,itm,set—1,oblk**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	None	

Description

The value of the item data item (field) in the current member of **set-1** is returned in the host language variable of **oblk**. The type and size of this variable must be consistent with the type and size of **itm** in the DDL specification. A nonzero command status is returned if a user does not have read access to the **itm** field.

Examples of Command Usage

```
block/direct   .. E0 = GFM ("itm,set-1,oblk")
block/indirect .. E0 = DMS ("GFM,itm,set-1,oblk")
record/direct  .. E0 = GFM ("itm,set-1",oblk)
record/indirect.. E0 = DMSD ("GFM,itm,set-1",oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

GFO — Get Field from Owner

GFO

Command Arguments **GFO,itm,set−1,oblk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	None	

Description

The value of the item data item (field) in the current owner of **set-1** is returned in the host language variable of **oblk**. The type and size of this variable must be consistent with the type and size of **itm** in the DDL specification. A nonzero command status is returned if a user does not have read access to the **itm** field.

Examples of Command Usage

```
block/direct   .. E0 = GFO ("itm,set-1,oblk")
block/indirect .. E0 = DMS ("GFO,itm,set-1,oblk")
record/direct  .. E0 = GFO ("itm,set-1",oblk)
record/indirect.. E0 = DMSD ("GFO,itm,set-1",oblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

ODRK — Obtain Duplicate Record based on calc Key — ODRK

Command Arguments ODRK,rec,blk

Currency Indicators

Used	Changed	Change Description
CRU	CRU	record with duplicate calc key value

Description

The types, sizes and sequence of host language variables designated for **blk** must be consistent with the data items defined for **rec**. Before ODRK is invoked, the desired calc key value should be assigned to the **blk** variable(s) that corresponds to the calc key's data item(s).

If the current of run unit is not null when ODRK is invoked:
Obtain an occurrence of the same record type as the current of run unit which has a calc key value that matches the calc key value held in **blk**'s variable(s). If such a record exists, it becomes the new current of run unit and the obtained data values are returned in the host language variables of **blk**. This command is valid only if the record that is the current of run unit is an occurrence of **rec** and **rec** has been defined (in the DDL) as having a calc key. The calc key value indicated by **blk** must duplicate the calc key value of the current of run unit; otherwise a nonzero command status results, no record is obtained and the current of run unit is not altered.

Use of ORK, followed by repeated use of ODRK, will obtain all records of a given record type having the same calc key value. The order in which these are obtained is not guaranteed. If an ODRK command is given when there are no more duplicates of a particular calc key value, then the command status is 255, no record is obtained, and the current of run unit becomes null.

If a user does not have read access to all of **rec**'s data items, then a nonzero command status results. The record is not obtained, but it does become the current of run unit.

If the current of run unit is null when ODRK is invoked:
ODRK has exactly the same effect as ORK.

Examples of Command Usage

```
block/direct   .. E0 = ODRK ("rec,blk")
block/indirect .. E0 = DMS ("ODRK,rec,blk")
record/direct  .. E0 = ODRK ("rec",blk)
record/indirect.. E0 = DMSD ("ODRK,rec",blk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

OFM **Obtain First Member** **OFM**

Command Arguments **OFM,set−1,oblk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	first member (set-1)
CRU		first member (set-1)

Description

The first member connected to the current owner of **set-1** is obtained. It becomes the current member of **set-1** and the current of the run unit. Its data values are returned in the host language variables of **oblk**. The types, sizes and sequence of variables designated for **oblk** must conform to the data items existing for the member record type. Which member is first depends on the **set-1** member order, as specified with the DDL.

If the current owner of **set-1** has no member, then the command status is 255, no record is obtained, the current of run unit becomes null, and the current member of **set-1** is null. If a user does not have read access to **set-1**, then a nonzero command status is returned, no record is obtained, and the currency indicators are unchanged. If a user has read access to **set-1** but not to all data items of its member record type, then a nonzero command status results. The currency indicators are changed, but no record is obtained.

Examples of Command Usage

```
block/direct   .. E0 = OFM ("set-1,oblk")
block/indirect .. E0 = DMS ("OFM,set-1,oblk")
record/direct  .. E0 = OFM ("set-1",oblk)
record/indirect.. E0 = DMSD ("OFM,set-1",oblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

OFO — Obtain First Owner — OFO

Command Arguments OFO,set-1,oblk

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	first owner (set-1)
	CRU	first owner (set-1)

Description

The first owner connected to the current member of **set-1** is obtained. It becomes the current owner of **set-1** and the current of the run unit. Its data values are returned in the host language variables of **oblk**. The types, sizes and sequence of variables designated for **oblk** must be consistent with the data items existing for the owner record type. Which owner is first depends on the **set-1** owner order, as specified with the DDL.

If the current member of **set-1** has no owner, then the command status is 255, no record is obtained, the current of run unit is null, and the current owner of **set-1** is null. If a user does not have read access to **set-1**, then a nonzero command status is returned, no record is obtained, and currency indicators are unchanged. If a user has read access to **set-1** but not to all data items of its owner record type, then a nonzero command status results. The currency indicators are changed, but no record is obtained.

Examples of Command Usage

```
block/direct   .. E0 = OFO ("set-1,oblk")
block/indirect .. E0 = DMS ("OFO,set-1,oblk")
record/direct  .. E0 = OFO ("set-1",oblk)
record/indirect.. E0 = DMSD ("OFO,set-1",oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

OLM — Obtain Last Member

OLM

Command Arguments

OLM,set−1,oblk

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	last member (set-1)
	CRU	last member (set-1)

Description

The last member connected to the current owner of **set-1** is obtained. It becomes the current member of **set-1** and the current of the run unit. Its data values are returned in the host language variables of **oblk**. The types, sizes and sequence of variables designated for **oblk** must be consistent with data items existing for the member record type. Which member is last depends on the **set-1** member order, as specified with the DDL.

If the current owner of **set-1** has no member, then the command status is 255, no record is obtained, the current of run unit becomes null and the current member of **set-1** becomes null. If a user does not have read access to **set-1**, then a nonzero command status is returned, no record is obtained, and the currency indicators are unchanged. If a user has read access to **set-1** but not to all data items of its member record type, then a nonzero command status results. The currency indicators are changed, but no record is obtained.

Examples of Command Usage

```
block/direct   .. E0 = OLM ("set-1,oblk")
block/indirect .. E0 = DMS ("OLM,set-1,oblk")
record/direct  .. E0 = OLM ("set-1",oblk)
record/indirect.. E0 = DMSD ("OLM,set-1",oblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

OLO — Obtain Last Owner — OLO

Command Arguments

OLO,set−1,oblk

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	last owner (set-1)
CRU	CRU	last owner (set-1)

Description

The last owner connected to the current member of **set-1** is obtained. It becomes the current owner of **set-1** and the current of the run unit. Its data values are returned in the host language variables of **oblk**. The types, sizes and sequence of variables designated for **oblk** must be consistent with data items existing for the owner record type. Which owner is last depends on the **set-1** owner order, as specified with the DDL.

If the current member of **set-1** has no owner, then the command status is 255, no record is obtained, the current of run unit becomes null, and the current owner of **set-1** becomes null. If a user does not have read access to **set-1**, then a nonzero command status is returned, no record is obtained, and no currency indicators are changed. If a user has read access to **set-1** but not to all data items of its owner record type, then a nonzero command status results. The currency indicators are changed, but no record is obtained.

Examples of Command Usage

```
block/direct   .. E0 = OLO ("set-1,oblk")
block/indirect .. E0 = DMS ("OLO,set-1,oblk")
record/direct  .. E0 = OLO ("set-1",oblk)
record/indirect.. E0 = DMSD ("OLO,set-1",oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

OMI — Obtain Member based on data Item

OMI

Command Arguments **OMI,item,set−1,blk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	first member (set-1) with data item value
CRU		first member (set-1) with data item value

Description

The types, sizes and sequence of host language variables designated for **blk** must be consistent with the data items existing for the member record type of **set-1**. Before OMI is invoked, the desired item value should be assigned to the **blk** variable that corresponds to **itm**. The system obtains the first member record connected to the current owner of **set-1**, and having the indicated value for **itm**. It becomes the current member of **set-1** and the current of run unit. The obtained data values are returned in the host language variables of **blk**. If there are two or more member records for the current owner of **set-1**, all having the **blk** value for **itm**, which one is first depends on the **set-1** member order. The command ONMI can be used to obtain the next member having the same value for **itm**.

If the value of **blk**'s host language variable does not match the **itm** value of any member record connected to the current owner of **set-1**, then the command status is 255, no record is obtained, the current member of **set-1** becomes null, and the current of run unit becomes null. If a user does not have read access to **set-1** and **itm**, then a nonzero command status will result, no record will be obtained, and currency indicators will remain unchanged. If a user has read access to **set-1** and **itm** but not to all data items of the member record type, then a nonzero command status results. The currency indicators are changed, but no record is obtained.

Examples of Command Usage

```
block/direct   .. E0 = OMI ("itm,set-1,blk")
block/indirect .. E0 = DMS ("OMI,itm,set-1,blk")
record/direct  .. E0 = OMI ("itm,set-1",blk)
record/indirect.. E0 = DMSD ("OMI,itm,set-1",blk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

OMSK — Obtain Member based on Sort Key

OMSK

Command Arguments **OMSK,set-1,blk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	first member (set-1) with sort key value
	CRU	first member (set-1) with sort key value

Description

Set-1 must have a sorted member order. The types, sizes and sequence of host language variables designated for **blk** must be consistent with the data items existing for the member record type of **set-1**. Before OMSK is invoked, the desired sort key value should be assigned to the **blk** variable(s) that corresponds to the sort key's data item(s). If RECORD-TYPE is part of the sort key, then OMSK cannot be used in place of FMSK and GETC. The system obtains the first member record connected to the current owner of **set-1** and having the indicated sort key value. This record becomes the current member of **set-1** and the current of run unit. Its data values are returned in the host language variables of **blk**.

If several member records for the current owner of **set-1** have the same sort key value, the one that is first depends on the DDL sort-clause specification for the **set-1** member order (either LIFO, FIFO, or IMMATERIAL).

If the current owner of **set-1** has no member whose sort key value matches the sort key value of **blk**, then the command status is 255 and no record is obtained. The member whose sort key value immediately follows the **blk** sort key value becomes the current member of **set-1**, as well as the current of run unit. The exceptions are when the current owner of **set-1** has no members or when **blk**'s sort key value is greater than the sort key value of the last member of **set-1**'s current owner. In these cases, the current member of **set-1** becomes null and the current of run unit becomes null.

A nonzero command status is returned if a user does not have read access to **set-1** and all data items that make up the member sort key. No record is obtained and currency indicators are unchanged. If a user has read access to **set-1** and the sort key but not to all data items of the member record type, then a nonzero command status is returned. Currency indicators are changed, but no record is obtained.

Examples of Command Usage

```
block/direct   .. E0 = OMSK ("set-1,blk")
block/indirect .. E0 = DMS ("OMSK,set-1,blk")
record/direct  .. E0 = OMSK ("set-1",blk)
record/indirect.. E0 = DMSD ("OMSK,set-1",blk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

ONM — Obtain Next Member

ONM

Command Arguments

ONM,set−1,oblk

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	next member (set-1)
CM(set-1)	CRU	next member (set-1)

Description

If the current member of **set-1** is not null when ONM is invoked:
The next member record (connected to the current owner of **set-1**) following the current member of **set-1** is obtained. It becomes the new current member of **set-1** and the current of run unit. Its data values are returned in the host language variables of **oblk**. The types, sizes and sequence of variables designated for **oblk** must be consistent with data items existing for the member record type. Which member record is logically next depends on the **set-1** member order, as specified with the DDL.

If the current member of **set-1** is the last member for the current owner of **set-1**, then ONM cannot find a next member. When there is no next member for the current owner of **set-1**, the command status is 255, no record is obtained, the current member of **set-1** becomes null, and the current of run unit becomes null. If a user does not have read access to **set-1**, then a nonzero command status is returned, no record is obtained, and the currency indicators remain unchanged. If a user has read access to **set-1** but not to all data items of its member record type, then a nonzero command status results. The currency indicators are changed, but no record is obtained.

If the current member of **set-1** is null when ONM is invoked:
ONM has exactly the same effect as OFM.

Examples of Command Usage

```
block/direct    .. E0 = ONM ("set-1,oblk")
block/indirect .. E0 = DMS ("ONM,set-1,oblk")
record/direct   .. E0 = ONM ("set-1",oblk)
record/indirect.. E0 = DMSD ("ONM,set-1",oblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

ONMI — Obtain Next Member based on data Item — ONMI

Command Arguments ONMI,itm,set−1,blk

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	next member (set-1) with item value
CM(set-1)	CRU	next member (set-1) with item value

Description

The types, sizes and sequence of host language variables designated for **blk** must be consistent with the data items existing for the member record type of **set-1**. Before ONMI is invoked, the desired item value should be assigned to the **blk** variable that corresponds to **itm**.

If the current member of **set-1** is not null when ONMI is invoked:
The system obtains the next member record connected to the current owner of **set-1** and having the indicated value for **itm**. This record becomes the current member of **set-1** and the current of run unit. The obtained data values are returned in the host language variables of **blk**. If there are two or more member records following the current member of **set-1**, all having the **blk** value for **itm**, which one is next depends on the **set-1** member order. If there is no next member with the indicated value for **itm**, the command status is 255, no record is obtained, the current member of **set-1** becomes null, and the current of run unit becomes null. If a user does not have read access to both **itm** and **set-1**, then a nonzero command status is returned, no record is obtained, and currency indicators remain unchanged. If a user has read access to **set-1** and **itm** but not to all member data items, then a nonzero command status results. Currency indicators are changed, but no data values are retrieved.

If the current member of **set-1** is null when ONMI is invoked:
ONMI has exactly the same effect as OMI. The first member record connected to the current owner of **set-1**, and having a value for **itm** that matches the indicated **blk** value becomes the current member of **set-1**.

Examples of Command Usage

```
block/direct   .. E0 = ONMI ("itm,set-1,blk")
block/indirect .. E0 = DMS ("ONMI,itm,set-1,blk")
record/direct  .. E0 = ONMI ("itm,set-1",blk)
record/indirect.. E0 = DMSD ("ONMI,itm,set-1",blk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

ONMSK — Obtain Next Member based on Sort Key

ONMSK

Command Arguments ONMSK,set-1,blk

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	next member (set-1) with sort key value
CM(set-1)	CRU	next member (set-1) with sort key value

Description

Set-1 must have a sorted member order. The type, sizes and sequence of host language variables designated for **blk** must be consistent with the data items existing for the member record type of **set-1**. Before ONMSK is invoked, the desired sort key value should be assigned to the **blk** variable(s) that corresponds to the sort key's data item(s). If RECORD-TYPE is part of the sort key, then ONMSK cannot be used in place of FNMSK and GETC. The system obtains the next member record (after the current member) that has the indicated sort key value and is connected to the current owner of **set-1**. This record becomes the current member of **set-1** and the current of run unit. Its data values are returned in the host language variables of **blk**.

If **set-1** has no current member, then ONMSK behaves just as OMSK. If several member records for the current owner of **set-1** have the same sort key value, the one that is next depends on the DDL sort-clause specification for the **set-1** member order (either LIFO, FIFO, or IMMATERIAL).

If the current owner of **set-1** has no member after the current member whose sort key value matches the sort key value of **blk**, then the command status is 255 and no record is obtained. The next member immediately following the current member becomes the current member of **set-1**, as well as the current of run unit. The exceptions are when the current owner of **set-1** has no members or when **blk**'s sort key value is greater than the sort key value of the last member of **set-1**'s current owner. In these cases, the current member of **set-1** becomes null and the current of run unit becomes null.

A nonzero command status is returned if a user does not have read access to **set-1** and all data items that make up the member sort key. No record is obtained and currency indicators are unchanged. If a user has read access to **set-1** and the sort key but not to all member data items, then a nonzero command status results. The currency indicators are changed, but no data values are retrieved.

Examples of Command Usage

```
block/direct   .. E0 = ONMSK ("set-1,blk")
block/indirect .. E0 = DMS ("ONMSK,set-1,blk")
record/direct  .. E0 = ONMSK ("set-1",blk)
record/indirect.. E0 = DMSD ("ONMSK,set-1",blk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

ONO — Obtain Next Owner — ONO

Command Arguments

ONO,set−1,oblk

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	next owner (set-1)
CO(set-1)	CRU	next owner (set-1)

Description

If the current owner of **set-1** is not null when ONO is invoked:
The next owner record (connected to the current member of **set-1**) following the current owner of **set-1** is obtained. This record becomes the new current owner of **set-1** and the current of run unit. Its data values are returned in the host language variables of **oblk**. The types, sizes and sequence of variables designated for **oblk** must be consistent with data items existing for the owner record type. Which owner record is logically next depends on the **set-1** owner order, as specified with the DDL.

If the current owner of **set-1** is the last owner for the current member of **set-1**, then ONO cannot find a next owner. When there is no next owner for the current member of **set-1**, the command status is 255, no record is obtained, the current owner of **set-1** becomes null, and the current of run unit becomes null. If a user does not have read access to **set-1**, then a nonzero command status is returned, no record is obtained, and currency indicators remain unchanged. If a user has read access to **set-1** but not to all owner data items, then a nonzero command status results. Currency indicators are changed, but no data values are retrieved.

If the current owner of **set-1** is null when ONO is invoked:
ONO has exactly the same effect as OFO.

Examples of Command Usage

```
block/direct   .. E0 = ONO ("set-1,oblk")
block/indirect .. E0 = DMS ("ONO,set-1,oblk")
record/direct  .. E0 = ONO ("set-1",oblk)
record/indirect.. E0 = DMSD ("ONO,set-1",oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

ONOI — Obtain Next Owner based on data Item

ONOI

Command Arguments **ONOI,itm,set−1,blk**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	next owner(set-1) with item value
CO(set-1)	CRU	next owner(set-1) with item value

Description

The types, sizes and sequence of host language variables designated for **blk** must be consistent with the data items existing for the owner record type of **set-1**. Before ONOI is invoked, the desired item value should be assigned to the **blk** variable that corresponds to **itm**.

If the current owner of **set-1** is not null when ONOI is invoked:
The system obtains the next owner record connected to the current member of **set-1** and having the indicated value for **itm**. This record becomes the current owner of **set-1** and the current of run unit. The obtained data values are returned in the host language variable of **blk**. If there are two or more owner records following the current owner of **set-1**, all having the **blk** value for **itm**, which one is next depends on the **set-1** owner order. If there is no next owner with the indicated value for **itm**, then the command status is 255, no record is obtained, the current owner of **set-1** becomes null, and the current of run unit becomes null. If a user does not have read access to both **itm** and **set-1**, then a nonzero command status is returned, no record is obtained, and currency indicators remain unchanged. If a user has read access to **set-1** and **itm** but not to all owner data items, then a nonzero command status results. Currency indicators are changed, but no data are retrieved.

If the current owner of **set-1** is null when ONOI is invoked:
ONOI has exactly the same effect as OOI. The first owner record connected to the current member of **set-1**, and having a value for **itm** that matches the indicated **blk** value becomes the current owner of **set-1**.

Examples of Command Usage

```
block/direct    .. E0 = ONOI ("itm,set-1,blk")
block/indirect .. E0 = DMS ("ONOI,itm,set-1,blk")
record/direct   .. E0 = ONOI ("itm,set-1",blk)
record/indirect.. E0 = DMSD ("ONOI,itm,set-1",blk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

ONOSK **Obtain Next Owner based on Sort Key** **ONOSK**

Command Arguments **ONOSK,set-1,blk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CO(set-1)	next owner (set-1) with sort key value
CM(set-1)	CRU	next owner (set-1) with sort key value

Description

Set-1 must have a sorted owner order. The types, sizes and sequence of host language variables designated for **blk** must be consistent with the data items existing for the owner record type of **set-1**. Before ONOSK is invoked, the desired sort key value should be assigned to the **blk** variable(s) that corresponds to the sort key's data item(s). If RECORD-TYPE is a part of the sort key, then ONOSK cannot be used in place of FNOSK and GETC. The system obtains the next owner record after the current owner that has the indicated sort key value and is connected to the current member of **set-1**. This record becomes the current owner of **set-1** and the current of the run unit. Its data values are returned in the host language variables of **blk**.

If **set-1** has no current owner, then ONOSK behaves just as OOSK. If several owner records for the current member of **set-1** have the same sort key value, the one that is next depends on the DDL sort-clause specification for the **set-1** owner order (either LIFO, FIFO, or IMMATERIAL).

If the current member of **set-1** has no owner after the current owner whose sort key value matches the sort key value of **blk**, then the command status is 255 and no record is obtained. The next owner immediately following the current member becomes the current owner of **set-1**, as well as the current of run unit. The exceptions are when the current member of **set-1** has no owners or when **blk**'s sort key value is greater than the sort key value of the last owner of **set-1**'s current member. In these cases, the current owner of **set-1** becomes null and the current of run unit becomes null.

A nonzero command status is returned if a user does not have read access to **set-1** and all data items that make up the owner sort key. No record is obtained and currency indicators are unchanged. If a user has read access to **set-1** and the sort key but not to all owner data items, then a nonzero command status results. Currency indicators are changed, but no data are retrieved.

Examples of Command Usage

```
block/direct    .. E0 = ONOSK ("set-1,blk")
block/indirect .. E0 = DMS ("ONOSK,set-1,blk")
record/direct   .. E0 = ONOSK ("set-1",blk)
record/indirect.. E0 = DMSD ("ONOSK,set-1",blk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

OOI — Obtain Owner based on data Item

OOI

Command Arguments **OOI,item,set−1,blk**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	first owner (set-1) with data item value
CRU		first owner (set-1) with data item value

Description

The types, sizes and sequence of host language variables designated for **blk** must be consistent with the data items existing for the owner record type of **set-1**. Before OOI is invoked, the desired item value should be assigned to the **blk** variable that corresponds to **itm**. The system obtains the first owner record connected to the current member of **set-1** and having the indicated value for **itm**. This record becomes the current owner of **set-1** and the current of run unit. The obtained data values are returned in the host language variables of **blk**. If there are two or more owner records for the current member of **set-1**, all having the **blk** value for **itm**, which one is first depends on the **set-1** owner order. The command ONOI can be used to obtain the next owner having the same value for **itm**.

If the value of **blk**'s host language variable for **itm** does not match the **itm** value of any owner record connected to the current member of **set-1**, then the command status is 255, no record is obtained, the current owner of **set-1** becomes null, and the current of run unit becomes null. If a user does not have read access to both **itm** and **set-1**, then a nonzero command status will result, no record will be obtained, and currency indicators will remain unchanged. If a user has read access to **set-1** and **itm** but not to all owner data items, then a nonzero command status results. Currency indicators are changed, but no data are retrieved.

Examples of Command Usage

```
block/direct   .. E0 = OOI ("itm,set-1,blk")
block/indirect .. E0 = DMS ("OOI,itm,set-1,blk")
record/direct  .. E0 = OOI ("itm,set-1",blk)
record/indirect.. E0 = DMSD ("OOI,itm,set-1",blk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

OOSK — Obtain Owner based on Sort Key OOSK

Command Arguments **OOSK,set-1,blk**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	first owner (set-1) with sort key value
	CRU	first owner (set-1) with sort key value

Description

Set-1 must have a sorted owner order. The types, sizes and sequence of host language variables designated for **blk** must be consistent with the data items existing for the owner record type of **set-1**. Before OOSK is invoked, the desired sort key value should be assigned to the **blk** variable(s) that corresponds to the sort key's data item(s). If RECORD-TYPE is part of the sort key, then OOSK cannot be used in place of FOSK and GETC. The system obtains the first owner record connected to the current member of **set-1** and having the indicated sort key value. This record becomes the current owner of **set-1** and the current of the run unit. Its data values are returned in the host language variables of **blk**.

If several owner records for the current member of **set-1** have the same sort key value, the one that is first depends on the DDL sort-clause specification for the **set-1** owner order (either LIFO, FIFO, or IMMATERIAL).

If the current member of **set-1** has no owner whose sort key value matches the sort key value of **blk**, then the command status is 255 and no record is obtained. The owner whose sort key value immediately follows the **blk** sort key value becomes the current owner of **set-1**, as well as the current of run unit. The exceptions are when the current member of **set-1** has no owners or when **blk**'s sort key value is greater than the sort key value of the last owner of **set-1**'s current member. In these cases, the current owner of **set-1** becomes null and the current of run unit becomes null.

A nonzero command status is returned if a user does not have read access to **set-1** and all data items that make up the owner sort key. No record is obtained and currency indicators are unchanged. If a user has read access to **set-1** and the sort key but not to all owner data items, then a nonzero command status results. Currency indicators are changed, but no data are retrieved.

Examples of Command Usage

```
block/direct   .. E0 = OOSK ("set-1,blk")
block/indirect .. E0 = DMS ("OOSK,set-1,blk")
record/direct  .. E0 = OOSK ("set-1",blk)
record/indirect.. E0 = DMSD ("OOSK,set-1",blk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

OPM — Obtain Prior Member — OPM

Command Arguments

OPM,set−1,oblk

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	prior member(set-1)
CM(set-1)	CRU	prior member(set-1)

Description

If the current member of **set-1** is not null when OPM is invoked:
The member record (connected to the current owner of **set-1**) preceding the current member of **set-1** is obtained. This record becomes the new current member of **set-1** and the current of run unit. Its data values are returned in the host language variables of **oblk**. The types, sizes and sequence of variables designated for **oblk** must be consistent with data items existing for the member record type. Which member record is logically prior depends on the **set-1** member order, as specified with the DDL.

If the current member of **set-1** is the first member for the current owner of **set-1**, then OPM cannot find a prior member. When there is no prior member for the current owner of **set-1**, then the command status is 255, the current member of **set-1** becomes null, and the current of run unit becomes null. If a user does not have read access to **set-1**, then a nonzero command status is returned and the currency indicators remain unchanged. If a user has read access to **set-1** but not to all member data items, then a nonzero command status results. Currency indicators are changed, but no data are retrieved.

If the current member of **set-1** is null when OPM is invoked:
OPM has exactly the same effect as OLM.

Examples of Command Usage

```
block/direct   .. E0 = OPM ("set-1,oblk")
block/indirect .. E0 = DMS ("OPM,set-1,oblk")
record/direct  .. E0 = OPM ("set-1",oblk)
record/indirect.. E0 = DMSD ("OPM,set-1",oblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

OPMI — Obtain Prior Member based on data Item — OPMI

Command Arguments OPMI,itm,set−1,blk

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	prior member (set-1) with item value
CM(set-1)	CRU	prior member (set-1) with item value

Description

The types, sizes and sequence of host language variables designated for **blk** must be consistent with the data items existing for the member record type of **set-1**. Before OPMI is invoked, the desired item value should be assigned to the **blk** variable that corresponds to **itm**.

If the current member of **set-1** is not null when OPMI is invoked:
The system obtains the member record connected to the current owner of **set-1** that precedes the current member and has the indicated value for **itm**. This record becomes the current member of **set-1** and the current of run unit. The obtained data values are returned in the host language variable of **blk**. If there are two or more member records preceding the current member of **set-1**, all having the **blk** value for **itm**, which one is prior depends on the **set-1** member order. If there is no prior member with the indicated value for **itm**, then the command status is 255, no record is obtained, the current member of **set-1** becomes null, and the current of run unit becomes null. If a user does not have read access to both **itm** and **set-1**, then a nonzero command status is returned, no record is obtained, and currency indicators remain unchanged. If a user has read access to **itm** and **set-1** but not to all member data items, then a nonzero command status results. Currency indicators are changed, but no data are retrieved.

If the current member of **set-1** is null when OPMI is invoked:
The last member record connected to the current owner of **set-1**, and having a value for **itm** that matches the indicated **blk** value becomes the current member of **set-1**.

Examples of Command Usage

```
block/direct   .. E0 = OPMI ("itm,set-1,blk")
block/indirect .. E0 = DMS ("OPMI,itm,set-1,blk")
record/direct  .. E0 = OPMI ("itm,set-1",blk)
record/indirect.. E0 = DMSD ("OPMI,itm,set-1",blk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

OPMSK — Obtain Prior Member based on Sort Key — OPMSK

Command Arguments

OPMSK,set-1,blk

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	prior member (set-1) with sort key value
CM(set-1)	CRU	prior member (set-1) with sort key value

Description

Set-1 must have a sorted member order. The types, sizes and sequence of host language variables designated for **blk** must be consistent with the data items existing in the member record type for **set-1**. Before OPMSK is invoked, the desired sort key value should be assigned to the **blk** variable(s) that corresponds to the sort key's data item(s). If RECORD-TYPE is part of the sort key, then OPMSK cannot be used in place of FPMSK and GETC. The member record prior to the current member that has the sort key value of **blk** and that is connected to the current owner of **set-1** becomes the current member of **set-1** and the current of the run unit.

If **set-1** has no current member, then the last member having the sort key value is found. If several member records for the current owner of **set-1** have the same sort key value, the one that is prior depends on the DDL sort-clause specification for the **set-1** member order (either LIFO, FIFO, or IMMATERIAL).

If the current owner of **set-1** has no member before the current member whose sort key value matches the sort key value of **blk**, then the command status is 255 and no record is obtained. The member immediately preceding the current member becomes the current member of **set-1**, as well as the current of run unit. The exceptions are when the current owner of **set-1** has no members or when **blk**'s sort key value is lower than the sort key value of the first member of **set-1**'s current owner. In these cases, the current member of **set-1** becomes null and the current of run unit becomes null.

A nonzero command status is returned if a user does not have read access to **set-1** and all data items that make up the member sort key. No record is obtained and currency indicators remain unchanged. If a user has read access to **set-1** and the sort key but not to all member data items, then a nonzero command status results. Currency indicators are changed, but no data are retrieved.

Examples of Command Usage

```
block/direct   .. E0 = OPMSK ("set-1,blk")
block/indirect .. E0 = DMS ("OPMSK,set-1,blk")
record/direct  .. E0 = OPMSK ("set-1",blk)
record/indirect.. E0 = DMSD ("OPMSK,set-1",blk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

OPO — Obtain Prior Owner — OPO

Command Arguments OPO,set-1,oblk

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	prior owner (set-1)
CO(set-1)	CRU	prior owner (set-1)

Description

If the current owner of **set-1** is not null when OPO is invoked:
The owner record (connected to the current member of **set-1**) preceding the current owner of **set-1** is obtained. This record becomes the new current owner of **set-1** and the current of run unit. Its data values are returned in the host language variables of **oblk**. The types, sizes and sequence of variables designated for **oblk** must be consistent with data items existing for the owner record type. Which owner record is logically prior depends on the **set-1** owner order, as specified with the DDL.

If the current owner of **set-1** is the first owner for the current member of **set-1**, then OPO cannot find a prior owner. When there is no prior owner for the current member of **set-1**, then the command status is 255, no record is obtained, the current owner of **set-1** becomes null, and the current of run unit becomes null. If a user does not have read access to **set-1**, then a nonzero command status is returned and currency indicators remain unchanged. If a user has read access to **set-1** but not to all its owner data items, then a nonzero command status results. Currency indicators are changed, but no data are retrieved.

If the current owner of **set-1** is null when OPO is invoked:
OPO has exactly the same effect as OLO.

Examples of Command Usage

```
block/direct   .. EO = OPO ("set-1,oblk")
block/indirect .. EO = DMS ("OPO,set-1,oblk")
record/direct  .. EO = OPO ("set-1",oblk)
record/indirect.. EO = DMSD ("OPO,set-1",oblk)
record/indirect.. E0 = DMS ("FPO,set-1")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

OPOI — Obtain Prior Owner based on data Item — OPOI

Command Arguments

OPOI,itm,set−1,blk

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	prior owner (set-1) with item value
CO(set-1)	CRU	prior owner (set-1) with item value

Description

The type, sizes and sequences of host language variables designated for **blk** must be consistent with the data items existing for the owner record type of **set-1**. Before OPOI is invoked, the desired item value should be assigned to the **blk** variable that corresponds to **itm**.

If the current owner of **set-1** is not null when OPOI is invoked:
The system obtains the prior owner record connected to the current member of **set-1** which has the indicated value for **itm**. This record becomes the current owner of **set-1** and the current of run unit. The obtained data values are returned in the host language variable of **blk**. If there are two or more owner records preceding the current owner of **set-1**, all having the **blk** value for **itm**, which one is prior depends on the **set-1** owner order. If there is no prior owner with the indicated value for **itm**, then the command status is 255, no record is obtained, the current owner of **set-1** becomes null, and the current of run unit becomes null. If a user does not have read access to both **itm** and **set-1**, then a nonzero command status is returned, no record is obtained, and currency indicators remain unchanged. If a user has read access to **itm** and **set-1**, but not to all owner data items, then a nonzero command status results. Currency indicators are changed, but no data are retrieved.

If the current owner of **set-1** is null when OPOI is invoked:
The last owner record connected to the current member of **set-1**, and having a value for **itm** that matches the indicated **blk** value becomes the current owner of **set-1**.

Examples of Command Usage

```
block/direct   .. E0 = OPOI ("itm,set-1,blk")
block/indirect .. E0 = DMS ("OPOI,itm,set-1,blk")
record/direct  .. E0 = OPOI ("itm,set-1",blk)
record/indirect.. E0 = DMSD ("OPOI,itm,set-1",blk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

OPOSK — Obtain Prior Owner based on Sort Key

OPOSK

Command Arguments **OPOSK,set-1,blk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CO(set-1)	prior owner (set-1) with sort key value
CM(set-1)	CRU	prior owner (set-1) with sort key value

Description

Set-1 must have a sorted owner order. The types, sizes and sequence of host language variables designated for **blk** must be consistent with the data items existing for the owner record type of **set-1**. Before OPOSK is invoked, the desired sort key value should be assigned to the **blk** variable(s) that corresponds to the sort key's data item(s). If RECORD-TYPE is a part of the sort key, then OPOSK cannot be used in place of FPOSK and GETC. The system obtains the prior owner record (before the current owner) that has the indicated sort key value and is connected to the current member of **set-1**. This record becomes the current owner of **set-1** and the current of the run unit. Its data values are returned in the host language variables of **blk**.

If **set-1** has no current owner, then the last owner having the sort key value is found. If several owner records for the current member of **set-1** have the same sort key value, the one that is prior depends on the DDL sort-clause specification for the **set-1** owner order (either LIFO, FIFO, or IMMATERIAL).

If the current member of **set-1** has no owner before the current owner whose sort key value matches the sort key value of **blk**, then the command status is 255 and no record is obtained. The owner immediately preceding the current member becomes the current owner of **set-1**, as well as the current of run unit. The exceptions are when the current member of **set-1** has no owners or when **blk**'s sort key value is less than the sort key value of the first owner of **set-1**'s current member. In these cases, the current owner of **set-1** becomes null and the current of run unit becomes null.

A nonzero command status is returned if a user does not have read access to **set-1** and all data items that make up the owner sort key. No record is obtained and currency indicators are unchanged. If a user has read access to **set-1** and the sort key but not to all owner data items, then a nonzero command status results. Currency indicators are changed, but no data are retrieved.

Examples of Command Usage

```
block/direct   .. E0 = OPOSK ("set-1,blk")
block/indirect .. E0 = DMS ("OPOSK,set-1,blk")
record/direct  .. E0 = OPOSK ("set-1",blk)
record/indirect.. E0 = DMSD ("OPOSK,set-1",blk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Retrieval Commands

ORK — Obtain Record based on calc Key — ORK

Command Arguments **ORK,rec,blk**

Currency Indicators

Used	Changed	Change Description
none	CRU	record with calc key value

Description

This command is valid only if **rec** has a calc key clause in the DDL specification. The types, sizes and sequence of host language variables designated for **blk** must be consistent with the data items defined for **rec**. Before ORK is invoked, the desired calc key value should be assigned to the **blk** variable(s) that corresponds to the calc key's data item(s). The system obtains an occurrence of **rec** having the indicated calc key value. This record becomes the current of run unit.

Occurrences of **rec** with duplicate calc key values may exist in the data base if the DDL specification declares that duplicates are allowed. If there is more than one occurrence of **rec** whose calc key value matches the indicated value, then ORK obtains one of these duplicates and ODRK can be used repeatedly to obtain each of the other duplicates.

If there is no **rec** occurrence having the indicated calc key value, then the command status is 255, no record is obtained and CRU becomes null. A nonzero command status is returned if a user does not have read access to all data items that make up **rec**'s calc key. No record is obtained and currency indicators remain unchanged. If a user has read access to the calc key but not to all **rec** data items, then a nonzero command status results. The current of run unit changes, but no data are retrieved.

Examples of Command Usage

```
block/direct   .. E0 = ORK ("rec,blk")
block/indirect .. E0 = DMS ("ORK,rec,blk")
record/direct  .. E0 = ORK ("rec",blk)
record/indirect.. E0 = DMSD ("ORK,rec",blk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

V. MODIFY COMMANDS

MDBS III provides six DML commands to modify the data values of a data base record occurrence. The modify commands transfer data from the host language variables (a data block or program data structure) into a current data base record occurrence.

These commands do not change any currency indicator within the data base; they only use currency indicators to identify the specific record occurrence that is to be changed. The modify commands change the data values of a specific record occurrence that is the current owner of a set, the current member of a set, or the current of run unit.

The modify commands change a record occurrence in one of two ways:

- Put a new value into a single data item
- Put new values into all data items

In modifying a single data item, the put commands transfer the value of a particular data item from the appropriate host language program variable to the record occurrence. The commands that change a single data item provide maximum data independence offered by MDBS III.

In modifying an entire record occurrence, the put commands transfer values of all data items from the appropriate host language program variables to the record occurrence.

The host language variables must be consistent with the corresponding data base items in type and size. When modifying data values of an entire record occurrence, match the host language data structure (data block or program record) with the corresponding record type. Specifically, make sure each host language variable of the data structure corresponds in sequence with each data item in the record type (as defined in the DDL source text).

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Modify Commands

The data base control system automatically enforces data integrity conditions (defined in the DDL source specification) when it performs any modify command. If a new data item value is outside the feasibility range for the data item, MDBS.DMS returns the appropriate command status; the modification is not performed.

The data base control system also automatically enforces data security conditions (defined in the DDL source specification) when it performs any modify command. If an application program user attempts a put command without having the correct write access codes to make the change, MDBS.DMS returns the appropriate command status; the modification is not performed.

Effects Upon Records with Retrieval Keys

For sorted sets, MDBS.DMS automatically maintains sorted order within the set when a data value that is part of a sort key is modified.

The modify commands change data values for calc key record occurrences, as long as the data items comprising the calc key keep their original values. The data base control does not permit changing values for data items that participate in a calc key. To modify calc key data item values, a new record occurrence has to be created. Use the following steps to create the new record:

1. retrieve the record contents
2. modify the data values in the host language data structure
3. create a new record occurrence with the edited data values
4. establish relevant set connections for the new record
5. delete the original record from the data base

For multiple area data bases, all areas permitting pointer indices must be on-line while performing the modify commands.

If the data base control system returns any command status other than zero, the modify command does not transfer any data to the data base record occurrence. The record occurrence remains unchanged.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Modify Commands

COMMAND DETAILS

PFC **Put data into Field of Current of run unit** **PFC**

Command Arguments **PFC,itm,iblk**

Currency Indicators

Used	Changed	Change Description
CRU	None	

Description

The value of the **iblk** host language variable is put into the **itm** field of the record that is current of run unit. The type and size of this variable must be consistent with **itm**. A nonzero command status is returned if the user does not have write access to **itm**. A nonzero command status is also returned if the value of **iblk**'s variable is outside of **itm**'s feasibility range.

The data base control system will return the appropriate nonzero command status for any of the conditions:

- the user does not have write access to **itm**
- the value of the **iblk** variable is outside of the feasibility range defined for **itm**
- **itm** participates in a calc key

Examples of Command Usage

```
block/direct  .. E0 = PFC ("itm,iblk")
block/indirect .. E0 = DMS ("PFC,itm,iblk")
record/direct  .. E0 = PFC ("itm",iblk)
record/indirect.. E0 = DMSD ("PFC,itm",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Modify Commands

PFM **Put data into Field of Member** **PFM**

Command Arguments **PFM,itm,set−1,iblk**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	None	

Description

The value of the **iblk** host language variable is put into the **itm** field of the current member of **set-1**.

The data base control system will return the appropriate nonzero command status for any of the conditions:

- the user does not have write access to **itm**
- the value of the **iblk** variable is outside of the feasibility range defined for **itm**
- **itm** participates in a calc key

Examples of Command Usage

```
block/direct  .. E0 = PFM ("itm,set-1,iblk")
block/indirect .. E0 = DMS ("PFM,itm,set-1,iblk")
record/direct  .. E0 = PFM ("itm,set-1",iblk)
record/indirect.. E0 = DMSD ("PFM,itm,set-1",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Modify Commands

PFO — Put data into Field of Owner — PFO

Command Arguments **PFO,itm,set−1,iblk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	None	

Description

The value of the **iblk** host language variable is put into the **itm** field of the current owner of **set-1.**

The data base control system will return the appropriate nonzero command status for any of the conditions:

- the user does not have write access to **itm**
- the value of the **iblk** variable is outside of the feasibility range defined for **itm**
- **itm** participates in a calc key

Examples of Command Usage

```
block/direct    .. E0 = PFO ("itm,set-1,iblk")
block/indirect .. E0 = DMS ("PFO,itm,set-1,iblk")
record/direct   .. E0 = PFO ("itm,set-1",iblk)
record/indirect.. E0 = DMSD ("PFO,itm,set-1",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Modify Commands

PUTC **PUT data into Current of run unit** **PUTC**

Command Arguments **PUTC,iblk**

Currency Indicators

Used	Changed	Change Description
CRU	None	

Description

Data values of the **iblk** host language variables are put into the record that is the current of run unit. The types, sizes and sequence of these variables must be consistent with the data items that make up the record type for the current of run unit.

The data base control system will return the appropriate nonzero command status for any of the conditions:

- the user does not have write access to all data items that make up the record type for the current of run unit
- the value of an **iblk** variable is outside of the feasibility range defined for the corresponding data item.
- the value changed for a data item that participates as a calc key

Examples of Command Usage

```
block/direct   .. E0 = PUTC ("iblk")
block/indirect .. E0 = DMS ("PUTC,iblk")
record/direct  .. E0 = PUTC (iblk)
record/indirect.. E0 = DMSD ("PUTC",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Modify Commands

PUTM **PUT data into Member** **PUTM**

Command Arguments

PUTM,set-1,iblk

Currency Indicators

Used	Changed	Change Description
CM(set-1)	None	

Description

Data values of the **iblk** host language variables are put into the current member of **set-1**. The types, sizes and sequence of these variables must be consistent with the data items that make up the member record type for **set-1**.

The data base control system will return the appropriate nonzero command status for any of the conditions:

- the user does not have write access to all data items that make up the record type for the set member
- the value of an **iblk** variable is outside of the feasibility range defined for the corresponding data item.
- the value changed for a data item that participates as a calc key

Examples of Command Usage

```
block/direct   .. E0 = PUTM ("set-1,iblk")
block/indirect .. E0 = DMS ("PUTM,set-1,iblk")
record/direct  .. E0 = PUTM ("set-1",iblk)
record/indirect.. E0 = DMSD ("PUTM,set-1,"iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Modify Commands

PUTO **PUT data into Owner** **PUTO**

Command Arguments **PUTO,set−1,iblk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	None	

Description

Data values of the **iblk** host language variables are put into the current owner of **set-1**. The types, sizes and sequence of these variables must be consistent with the data items that make up the owner record type for **set-1**.

The data base control system will return the appropriate nonzero command status for any of the conditions:

- the user does not have write access to all data items that make up the record type for the set owner
- the value of an **iblk** variable is outside of the feasibility range defined for the corresponding data item.
- the value changed for a data item that participates as a calc key

Examples of Command Usage

```
block/direct   .. E0 = PUTO ("set-1,iblk")
block/indirect .. E0 = DMS ("PUTO,set-1,iblk")
record/direct  .. E0 = PUTO ("set-1",iblk)
record/indirect.. E0 = DMSD ("PUTO,set-1,"iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

VI. ASSIGNMENT COMMANDS

The Data Manipulation Language provides twenty-three commands that change currency indicator values. Specifically, the assignment commands change a currency indicator in the following ways:

- removes the record reference within a currency indicator
- copies the record reference within one currency indicator to another currency indicator

While the currency indicators do not contain a copy of record occurrences, they do identify individual records by record type and location within the data base. The assignment commands provide a second method (in addition to the find commands) to make records current in the data base. The assignment commands consist of four major groups that are based on the currency indicators they set (affect):

- Current of Run Unit - The set commands can assign the current of run unit to the following records:

  - the current owner of a particular set
  - the current member of a particular set
  - the current record of a user-defined currency indicator
  - a previously saved current of run unit
  - null (references no record)

  The current of run unit value can be saved in a host language program variable in the form of a data base key that identifies the location of the record within the data base. Once saved, a data base key can be transferred to the data base system and set as the value of the current of run unit.

- Set Member - The set commands can assign the set member currency indicators to the following records:

  - the current of run unit
  - the current owner of a particular set
  - the current member of a particular set
  - the current record of a user-defined currency indicator
  - null (references no record)

- Set Owner - The set commands can assign the set owner currency indicators to the following records:

  - the current of run unit
  - the current owner of a particular set
  - the current member of a particular set
  - the current record of a user-defined currency indicator
  - null (references no record)

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

- User-defined Indicators - The set commands can assign user-defined currency indicators to the following records:

  - the current of run unit
  - the current owner of a particular set
  - the current member of a particular set
  - the current record of a user-defined currency indicator
  - null (references no record)

Assignment Command Effects

Any assignment command that sets the current owner of set-1 to the value of another currency indicator (except the SOE and SON commands) also automatically locates the first member of set-1 (with a FFM command).

Any assignment command that sets the current member of set-1 to the value of another currency indicator (except the SME and SMN commands) also automatically locates the first owner of set-1 (with a FFO command).

The DBENV command can be used to suppress the FFM and FFO operations automatically following the assignment commands. See the DBENV in chapter twelve of this manual for additional information. When the automatic find command is suppressed, MDBS.DMS nulls the current set member currency indicator (for set owner commands) and the current set owner currency indicator (for set member commands). Setting a run unit's environment in this way may allow the program to be more compact if it needs to loop through owners or members of a set.

If a user does not have read access to all sets that are specified as arguments to an assignment command, then MDBS.DMS returns a nonzero command status to indicate the condition and no currency indicator is altered.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

Traversing Multiple Sets

By using the assignment commands together with the find commands, you can traverse (or locate) all the data base record occurrences accessible through set relationships.

For example, assume you need to retrieve the employee records for a particular department from a data base that consists of the following schema:

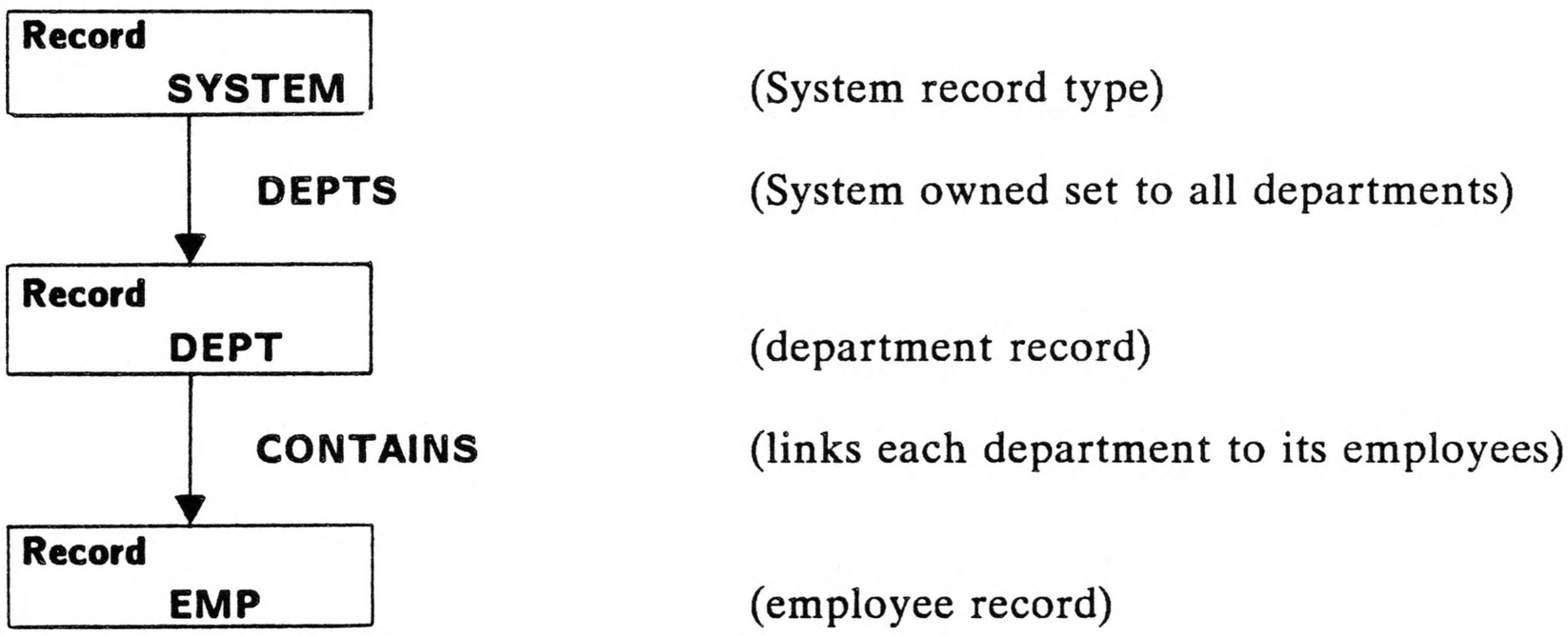

Figure VI-1. Example Schema Diagram

The employee records are connected to the set CONTAINS, so the application program needs to access the EMP records through this set. In this example, the application program needs to traverse the data base from the SYSTEM record to the appropriate EMP records; the four steps outlined below show you one method to accomplish this:

1. Find the correct department in the set DEPTS

Use one or more of the find commands to locate the particular department. For example, the Find Member based on data Item command finds the DEPT record whose department name (DNAME) matches the data value in iblk).

```
FMI ("DNAME, DEPTS", iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

2. Make this department the owner of the set CONTAINS

Use an assignment command to copy the value of one currency indicator to another. The command Set Owner based on Member command copies the value of the current member of DEPTS to the current owner of CONTAINS.

SOM ("CONTAINS, DEPTS")

To keep the syntax of this command straight, think of this command as a programming statement:

Current_Owner (CONTAINS) = Current_Member (DEPTS)

3. Find and retrieve the first employee record of the department

Use one of the retrieval commands to retrieve the first employee of the department. The command Obtain First Member finds and retrieves the first employee record connected to the current owner of DEPTS.

OFM ("CONTAINS", oblk)

4. Find and retrieve each remaining employee record of this department

Use another retrieval command to obtain each remaining employee record. The command Obtain Next Member finds and retrieves the next employee record in the department.

ONM("CONTAINS", oblk)

Repeat this command until you reach the end of the set.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

COMMAND DETAILS

SCD — Set Current of run unit to Data base key — SCD

Command Arguments

SCD,iblk

Currency Indicators

Used	Changed	Change Description
None	CRU	set to data base key

Description

The value of **iblk**'s host language variable becomes the new current of run unit. Normally, the value will be a data base key that was previously saved by the SDC command. If the system detects that the value is not a valid data base key, then command status 49 results for SCD and the current of run unit remains unchanged. The integrity of the data base itself will not be damaged.

In most environments, the **iblk** host language variable must be consistent with a four-byte binary data item. If some other type or size is necessary for a particular environment, then the difference is described in the pertinent system specific manual.

Examples of Command Usage

```
block/direct   .. EO = SCD ("iblk")
block/indirect .. EO = DMS ("SCD,iblk")
record/direct  .. EO = SCD (iblk)
record/indirect.. EO = DMSD ("SCD",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SCM **Set Current of run unit based on Member** SCM

Command Arguments **SCM,set-1**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CRU	set to CM(set-1)

Description

The record that is presently the current member of **set-1** becomes the new current of run unit. The current member of **set-1** remains unchanged. If the user does not have read access to **set-1**, a nonzero command status is returned and the current of run unit is unchanged.

Examples of Command Usage

```
block/direct   .. E0 = SCM ("set-1")
block/indirect .. E0 = DMS ("SCM,set-1")
record/direct  .. E0 = SCM ("set-1")
record/indirect.. E0 = DMS ("SCM,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SCN **Set Current of run unit to Null** SCN

Command Arguments

SCN (no arguments)

Currency Indicators

Used	Changed	Change Description
None	CRU	set to null

Description

The current of run unit becomes null.

Examples of Command Usage

```
block/direct   .. E0 = SCN (     )
block/indirect .. E0 = DMS ("SCN")
record/direct  .. E0 = SCN (     )
record/indirect.. E0 = DMS ("SCN")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SCO **Set Current of run unit based on Owner** SCO

Command Arguments **SCO,set–1**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CRU	set to CO(set-1)

Description

The record that is presently the current owner of **set-1** becomes the new current of run unit. The current owner of **set-1** remains unchanged. If the user does not have read access to **set-1**, a nonzero command status is returned and the current of run unit is unchanged.

Examples of Command Usage

```
block/direct   .. E0 = SCO ("set-1")
block/indirect .. E0 = DMS ("SCO,set-1")
record/direct  .. E0 = SCO ("set-1")
record/indirect.. E0 = DMS ("SCO,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SCU — Set Current of run unit based on User indicator — SCU

Command Arguments

SCU,iblk

Currency Indicators

Used	Changed	Change Description
CU(i)	CRU	set to CU(i)

Description

The current record of a user-defined indicator becomes the new current of run unit. The value of the host language variable for **iblk** determines which one of the user-defined indicators is involved in this command. The host language variable must be consistent with an unsigned data item that is one byte in size. The variable's value must be an integer between 1 and 255, inclusive. The current record of the user-defined indicator remains unchanged. The AUI command (Chapter XI) is employed to allocate user-defined indicators.

Examples of Command Usage

```
block/direct   .. E0 = SCU ("iblk")
block/indirect .. E0 = DMS ("SCU,iblk")
record/direct  .. E0 = SCU (iblk)
record/indirect.. E0 = DMSD ("SCU",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SDC **Set Data base key for Current of run unit** SDC

Command Arguments **SDC,oblk**

Currency Indicators

Used	Changed	Change Description
CRU	none	

Description

A data base key is a unique internal identifier for a record occurrence. The SDC command saves the data base key for the record that is the current of run unit. If the current of run unit is null, then a null data base key is saved. The data base key is saved in **oblk**'s host language variable. In most environments, this variable must be consistent with a four-byte binary data item. If some other type or size is necessary for a particular environment, then the difference is described in the pertinent system specific manual.

The SCD command can be used to set the current of run unit to be the same as a saved data base key.

Examples of Command Usage

```
block/direct    .. EO = SDC ("oblk")
block/indirect .. EO = DMS ("SDC,oblk")
record/direct  .. EO = SDC (oblk)
record/indirect.. EO = DMSD ("SDC",oblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SMC **Set Member based on Current of run unit** SMC

Command Arguments **SMC,set-1**

Currency Indicators

Used	Changed	Change Description
CRU	CM(set-1)	set to CRU
	CO(set-1)	set to first owner (set-1)
	CRU	set to first owner (set-1)

Description

The record that is the current of run unit becomes the new current member of **set-1**. The first owner connected to the new current member of **set-1** becomes the new current owner of **set-1** and the current of run unit. If the current member of **set-1** has no owners via **set-1**, then the current of run unit becomes null, the current owner of **set-1** becomes null, and a command status of 255 is returned. If the user does not have read access to **set-1**, a nonzero command status is returned and no currency indicators are changed.

Examples of Command Usage

```
block/direct   .. E0 = SMC ("set-1")
block/indirect .. E0 = DMS ("SMC,set-1")
record/direct  .. E0 = SMC ("set-1")
record/indirect.. E0 = DMS ("SMC,set-1")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SME — Set Member to current of run unit (Exception) — SME

Command Arguments **SME,set–1**

Currency Indicators

Used	Changed	Change Description
CRU	CM(set-1)	set to CRU

Description

The record that is the current of run unit becomes the new current member of **set-1**. The current of run unit remains unchanged. If a user does not have read access to **set-1**, the current member of **set-1** does not change and a nonzero command status is returned. If the record type of the current of run unit is not a member of **set-1**, then a nonzero command status is returned and the current member of **set-1** is unchanged. If a current owner for **set-1** exists and the record indicated by the current of run unit is not connected to that owner through **set-1**, then a command status of 12 is returned.

Examples of Command Usage

```
block/direct   .. E0 = SME ("set-1")
block/indirect .. E0 = DMS ("SME,set-1")
record/direct  .. E0 = SME ("set-1")
record/indirect.. E0 = DMS ("SME,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SMM — Set Member based on Member — SMM

Command Arguments SMM,set-1,set-2

Currency Indicators

Used	Changed	Change Description
CM(set-2)	CM(set-1)	set to CM(set-2)
	CO(set-1)	set to first owner (set-1)
	CRU	set to first owner (set-1)

Description

Set the current member of **set-1** to be the same as the current member of **set-2**. This command is valid only if **set-1** and **set-2** have the same member record type(s). The first owner of the new current member of **set-1** becomes the new current owner of **set-1** and the current of run unit. If the new current member of **set-1** has no owners via **set-1**, then the current owner of **set-1** becomes null, the current of run unit becomes null and a command status of 255 is returned. If a user does not have read access to both **set-1** and **set-2**, then a nonzero command status is returned and no currency indicators change.

Examples of Command Usage

```
block/direct    .. E0 = SMM ("set-1,set-2")
block/indirect .. E0 = DMS ("SMM,set-1,set-2")
record/direct   .. E0 = SMM ("set-1,set-2")
record/indirect.. E0 = DMS ("SMM,set-1,set-2")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SMN **Set Member to Null** SCN

Command Arguments **SMN,set−1**

Currency Indicators

Used	Changed	Change Description
	CM(set-1)	set to null

Description

The current member of **set-1** becomes null. The current owner of **set-1** is unaffected. If a user does not have read access to **set- 1**, then the current member of **set-1** does not change and a nonzero command status is returned.

Examples of Command Usage

```
block/direct   .. E0 = SMN ("set-1")
block/indirect .. E0 = DMS ("SMN,set-1")
record/direct  .. E0 = SMN ("set-1")
record/indirect.. E0 = DMS ("SMN,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SMO — Set Member based on Owner — SMO

Command Arguments **SMO,set-1,set-2**

Currency Indicators

Used	Changed	Change Description
CO(set-2)	CM(set-1)	set to CO(set-2)
	CO(set-1)	set to first owner (set-1)
	CRU	set to first owner (set-1)

Description

Set the current member of **set-1** to be the same as the current owner of **set-2**. This command is valid only if the member record type(s) for **set-1** is the same as the owner record type(s) for **set-2**. The first owner of the new current member of **set-1** becomes the new current owner of **set-1** and the current of run unit. If the new current member of **set-1** has no owners via set **set-1**, then the current owner of **set-1** becomes null, the current of run unit becomes null, and the command status is 255. If a user does not have read access to **set-1** and **set-2**, then a nonzero command status is returned and no currency indicators change.

Examples of Command Usage

```
block/direct    .. E0 = SMO ("set-1,set-2")
block/indirect .. E0 = DMS ("SMO,set-1,set-2")
record/direct   .. E0 = SMO ("set-1,set-2")
record/indirect.. E0 = DMS ("SMO,set-1,set-2")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SMU — Set Member based on User indicator — SMU

Command Arguments **SMU,set−1,iblk**

Currency Indicators

Used	Changed	Change Description
CU(i)	CM(set-1)	set to CU(i)
	CO(set-1)	set to first owner (set-1)
	CRU	set to first owner (set-1)

Description

Set the current member of **set-1** to be the same as the current record of a user-defined indicator. The value of the host language variable for **iblk** determines which one of the user-defined indicators is involved in this command. The host language variable must be consistent with an unsigned data item that is one byte in size. The variable's value must be an integer between 1 and 255, inclusive. The first owner of the new current member of **set-1** becomes the new current owner of **set-1** and the current of run unit. If the new current member has no owners via **set-1**, then the current owner of **set-1** becomes null, the current of run unit become null, and the command status is 255. The AUI command (Chapter XI) is employed to allocate user-defined indicators.

Examples of Command Usage

```
block/direct   .. E0 = SMU ("set-1,iblk")
block/indirect .. E0 = DMS ("SMU,set-1,iblk")
record/direct  .. E0 = SMU ("set-1",iblk)
record/indirect.. E0 = DMSD ("SMU,set-1",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SOC — Set Owner based on Current of run unit

SOC

Command Arguments **SOC,set–1**

Currency Indicators

Used	Changed	Change Description
CRU	CO(set-1)	set to CRU
	CM(set-1)	set to first member (set-1)
	CRU	set to first member (set-1)

Description

The record that is the current of run unit becomes the new current owner of **set-1**. The first member connected to the new current owner of **set-1** becomes the new current member of **set-1** and the current of run unit. If the current owner of **set-1** has no members via **set-1**, then the current of run unit becomes null, the current member of **set-1** becomes null, and a command status of 255 is returned. If the user does not have read access to **set-1**, a nonzero command status is returned and no currency indicators are changed.

Examples of Command Usage

```
block/direct    .. E0 = SOC ("set-1")
block/indirect .. E0 = DMS ("SOC,set-1")
record/direct   .. E0 = SOC ("set-1")
record/indirect.. E0 = DMS ("SOC,set-1")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SOE — Set Owner to current of run unit (Exception) — SOE

Command Arguments

SOE,set-1

Currency Indicators

Used	Changed	Change Description
CRU	CO(set-1)	set to CRU

Description

The record that is the current of run unit becomes the new current owner of **set-1**. The current of run unit remains unchanged. If a user does not have read access to **set-1**, the current owner of **set-1** does not change and a nonzero command status is returned. If the record type of the current of run unit is not an owner of **set-1**, then a nonzero command status is returned and the current owner of **set-1** is unchanged. If a current member for **set-1** exists and the record indicated by the current of run unit is not connected to that member through **set-1**, then a command status of 12 is returned.

Examples of Command Usage

```
block/direct   .. E0 = SOE ("set-1")
block/indirect .. E0 = DMS ("SOE,set-1")
record/direct  .. E0 = SOE ("set-1")
record/indirect.. E0 = DMS ("SOE,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SOM — Set Owner based on Member — SOM

Command Arguments SOM,set-1,set-2

Currency Indicators

Used	Changed	Change Description
CM(set-2)	CO(set-1)	set to CM(set-2)
	CM(set-1)	set to first member (set-1)
	CRU	set to first member (set-1)

Description

Set the current owner of **set-1** to be the same as the current member of **set-2**. This command is valid only if the owner record type(s) for **set-1** is the same as the member record type(s) for **set-2**. The first member of the new current owner of **set-1** becomes the new current member of **set-1** and the current of run unit. If the new current owner of **set-1** has no members via set **set-1**, then the current member of **set-1** becomes null, the current of run unit becomes null and the command status is 255. If a user does not have read access to **set-1** and **set-2**, then a nonzero command status is returned and no currency indicators change.

Examples of Command Usage

```
block/direct    .. E0 = SOM ("set-1,set-2")
block/indirect .. E0 = DMS ("SOM,set-1,set-2")
record/direct   .. E0 = SOM ("set-1,set-2")
record/indirect.. E0 = DMS ("SOM,set-1,set-2")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SON

Set Owner to Null

SON

Command Arguments **SON,set–1**

Currency Indicators

Used	Changed	Change Description
	CO(set-1)	set to null

Description

The current owner of **set-1** becomes null. The current member of **set-1** is unaffected. If a user does not have read access to **set-1**, then the current owner of **set-1** does not change and a nonzero command status is returned.

Examples of Command Usage

```
block/direct   .. E0 = SON ("set-1")
block/indirect .. E0 = DMS ("SON,set-1")
record/direct  .. E0 = SON ("set-1")
record/indirect.. E0 = DMS ("SON,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SOO — Set Owner based on Owner — SOO

Command Arguments

SOO,set-1,set-2

Currency Indicators

Used	Changed	Change Description
CO(set-2)	CO(set-1)	set to CO(set-2)
	CM(set-1)	set to first member (set-1)
	CRU	set to first member (set-1)

Description

Set the current owner of **set-1** to be the same as the current owner of **set-2**. This command is valid only if **set-1** and **set-2** have the same owner record type(s). The first member of the new current owner of **set-1** becomes the new current member of **set-1** and the current of run unit. If the new current owner of **set-1** has no members via **set-1**, then the current member of **set-1** becomes null, the current of run unit becomes null, and a command status of 255 is returned. If a user does not have read access to both **set-1** and **set-2**, then a nonzero command status is returned and no currency indicators change.

Examples of Command Usage

```
block/direct    .. E0 = SOO ("set-1,set-2")
block/indirect .. E0 = DMS ("SOO,set-1,set-2")
record/direct   .. E0 = SOO ("set-1,set-2")
record/indirect.. E0 = DMS ("SOO,set-1,set-2")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SOU — Set Owner based on User indicator

SOU

Command Arguments

SOU,set−1,iblk

Currency Indicators

Used	Changed	Change Description
CU(i)	CO(set-1)	set to CU(i)
	CM(set-1)	set to first member (set-1)
	CRU	set to first member (set-1)

Description

Set the current owner of **set-1** to be the same as the current record of a user-defined indicator. The value of the host language variable for **iblk** determines which one of the user-defined indicators is involved in this command. The host language variable must be consistent with an unsigned data item that is one byte in size. The variable's value must be an integer between 1 and 255, inclusive. The first member of the new current owner of **set-1** becomes the new current member of **set- 1** and the current of run unit. If the new current owner has no members via **set-1**, then the current member of **set-1** becomes null, the current of run unit becomes null, and the command status is 255. The AUI command (Chapter XI) is employed to allocate user-defined indicators.

Examples of Command Usage

```
block/direct   .. E0 = SOU ("set-1,iblk")
block/indirect .. E0 = DMS ("SOU,set-1,iblk")
record/direct  .. E0 = SOU ("set-1",iblk)
record/indirect.. E0 = DMSD ("SOU,set-1",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SUC — Set User indicator to Current of run unit — SUC

Command Arguments

SUC,iblk

Currency Indicators

Used	Changed	Change Description
CRU	CU(i)	set to CRU

Description

Set the current record of a user-defined indicator to be the same as the current of run unit. The value of the host language variable for **iblk** determines which one of the user-defined indicators is involved in this command. The host language variable must be consistent with an unsigned data item that is one byte in size. The variable's value must be an integer between 1 and 255, inclusive. The AUI command (Chapter XI) is employed to allocate user-defined indicators.

Examples of Command Usage

```
block/direct   .. E0 = SUC ("iblk")
block/indirect .. E0 = DMS ("SUC,iblk")
record/direct  .. E0 = SUC (iblk)
record/indirect.. E0 = DMSD ("SUC",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SUM

Set User indicator to Member

SUM

Command Arguments

SUM,set-1,iblk

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CU(i)	set to CM(set-1)

Description

Set the current record of a user-defined indicator to be the same as the current member of **set-1**. The value of the host language variable for **iblk** determines which one of the user-defined indicators is involved in this command. The host language variable must be consistent with an unsigned data item that is one byte in size. The variable's value must be an integer between 1 and 255, inclusive. The AUI command (Chapter XI) is employed to allocate user-defined indicators.

If a user does not have read access to **set-1**, no currency indicator changes and a nonzero command status is returned.

Examples of Command Usage

```
block/direct    .. E0 = SUM ("set-1,iblk")
block/indirect .. E0 = DMS ("SUM,set-1,iblk")
record/direct   .. E0 = SUM ("set-1",iblk)
record/indirect.. E0 = DMSD ("SUM,set-1",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SUN

Set User indicator to Null

SUN

Command Arguments **SUN,iblk**

Currency Indicators

Used	Changed	Change Description
none	CU(i)	set to null

Description

Set the current record of a user-defined indicator to be null. The value of the host language variable for **iblk** determines which one of the user-defined indicators is involved in this command. The host language variable must be consistent with an unsigned data item that is one byte in size. The variable's value must be an integer between 1 and 255, inclusive. The AUI command (Chapter XI) is employed to allocate user-defined indicators.

Examples of Command Usage

```
block/direct   .. E0 = SUN ("iblk")
block/indirect .. E0 = DMS ("SUN,iblk")
record/direct  .. E0 = SUN (iblk)
record/indirect.. E0 = DMSD ("SUN",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SUO — Set User indicator_to Owner

SUO

Command Arguments **SUO,set−1,iblk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CU(i)	set to CO(set-1)

Description

Set the current record of a user-defined indicator to be the same as the current owner of **set-1**. The value of the host language variable for **iblk** determines which one of the user-defined indicators is involved in this command. The host language variable must be consistent with an unsigned data item that is one byte in size. The variable's value must be an integer between 1 and 255, inclusive. The AUI command (Chapter XI) is employed to allocate user-defined indicators.

If a user does not have read access to **set-1**, no currency indicator changes and a nonzero command status is returned.

Examples of Command Usage

```
block/direct   .. E0 = SUO ("set-1,iblk")
block/indirect .. E0 = DMS ("SUO,set-1,iblk")
record/direct  .. E0 = SUO ("set-1",iblk)
record/indirect.. E0 = DMSD ("SUO,set-1",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Assignment Commands

SUU **Set User indicator to User indicator** SUU

Command Arguments **SUU,iblk**

Currency Indicators

Used	Changed	Change Description
CU(j)	CU(i)	set to CU(j)

Description

Two host language variables, say i and j, are defined for **iblk**. Each must be consistent with an unsigned data item that is one byte in size. Each must have an integer value between 1 and 255, inclusive. This command sets the current record of a user-defined indicator (determined by i's value) to be the same as the current record of another user-defined indicator (determined by j's value). The AUI command (Chapter XI) is employed to allocate user-defined indicators.

Examples of Command Usage

```
block/direct   .. E0 = SUU ("iblk")
block/indirect .. E0 = DMS ("SUU,iblk")
record/direct  .. E0 = SUU (iblk)
record/indirect.. E0 = DMSD ("SUU",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

VII. CREATION COMMANDS

OVERVIEW

To add new records in a data base, MDBS DML provides two create commands:

- create a record in a particular data base area
- create a new record anywhere in the data base

Data for the new record is transferred from the host language variables (a data block or program record type) to the data base. The host language variables must be consistent with the corresponding data base items in type and size. Make sure each host language variable of the data structure corresponds in sequence with each data item in the record type (as defined in the DDL source text). Data compression and encryption are automatic, where applicable.

The data base control system automatically enforces data integrity conditions (defined in the DDL source specification) when it creates a new record. If a new data item value is outside the feasibility range for the data item, MDBS.DMS returns the appropriate command status; the record is not created.

The data base control system also automatically enforces data security conditions (defined in the DDL source specification) when it creates a new record occurrence. If an application program user attempts to create a record without having the correct write access codes for all data items of the record type and all sets in which the record type participates as an automatic owner or member, MDBS.DMS returns the appropriate command status; the record is not created.

If a CALC or clustering record location mode is defined for the record, then MDBS.DMS physically positions the new record accordingly. For clustered records, MDBS.DMS locates the new record based on the current record of the set (current member or owner) specified for the record type in the DDL source specification.

The data base control system will not force a record to be created in an area not allowed by the DDL specification. Also, if there is no available space to create the new record occurrence, MDBS.DMS returns a nonzero command status to that effect; the record is not created.

If a data base has multiple areas, then all areas which permit pointer indices must be on-line during record creation.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Creation Commands

COMMAND DETAILS

CRA — Create Record in Area

CRA

Command Arguments **CRA,rec,area,iblk**

Currency Indicators

Used	Changed	Change Description
	CRU	set to new record occurrence

If the new record is automatically inserted into one or more sets, the data base control system uses and changes additional currency indicators according to the following table. See the notes listed below.

Used	Changed	Change Description
CO(set-1)	CM(set-1)	set to new record occurrence; **Note 1**
CM(set-2)	CO(set-2)	set to new record occurrence; **Note 2**

Description

An occurrence of the record type **rec** is created in the indicated **area**. The values of the host language variables for **iblk** are stored into that record occurrence.

The sequence, types, and sizes of host language variables defined for **iblk** must correspond to the sequence, types, and sizes of data items defined for **rec** in the DDL specification. The values of the variables must conform to feasibility ranges specified for the data items of the record type. If they do not, a nonzero command status is returned, the record is not created, and no currency indicators change.

The physical placement of the newly created record is consistent with the DDL specification for **rec** (either clustered or system-determined).

The CRA command will not create a record with a calc key.

In order to create a record with CRA, a user must have write access to **rec** and to all sets in which the record type participates as an automatic owner or member. If a user does not have this security clearance, a nonzero command status is returned, the record is not created, and no currency indicator is changed.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Creation Commands

Note 1 - If **rec** has been declared to be the AUTO member of set-1, the newly created record is automatically connected to the current owner of the set and the new record becomes the current member of the set. The connection takes place according to the member order specified in the DDL source text (such as SORTED or FIFO). If set-1 has no current owner, a nonzero command status is returned, the record is not created, and no currency indicators change.

Note 2 - If **rec** has been declared to be the AUTO owner of set-2, the newly created record is automatically connected to the current member of the set and the new record becomes the current owner of the set. The connection occurs according to the set's owner order, as specified with the DDL (such as SORTED or NEXT). If set-2 has no current member, a nonzero command status is returned, the record is not created, and no currency indicators change.

Examples of Command Usage

```
block/direct   .. E0 = CRA ("rec,area,iblk")
block/indirect .. E0 = DMS ("CRA,rec,area,iblk")
record/direct  .. E0 = CRA ("rec,area",iblk)
record/indirect.. E0 = DMSD ("CRA,rec,area",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Creation Commands

CRS **Create Record and Store** **CRS**

Command Arguments **CRS,rec,iblk**

Currency Indicators

Used	Changed	Change Description
	CRU	set to new record occurrence

If the new record is automatically inserted into one or more sets, the data base control system uses and changes additional currency indicators according to the following table. See the notes listed below.

Used	Changed	Change Description
CO(set-1)	CM(set-1)	set to new record occurrence; **Note 1**
CM(set-2)	CO(set-2)	set to new record occurrence; **Note 2**

Description

An occurrence of the record type **rec** is created in a permissible area. The values of the host language variables for **iblk** are stored into that record occurrence.

The sequence, types, and sizes of host language variables defined for **iblk** must correspond to the sequence, types, and sizes of data items defined for **rec** in the DDL specification. The values of the variables must conform to feasibility ranges specified for the record type's data items. If they do not, a nonzero command status is returned, the record is not created, and no currency indicators change.

The physical placement of the newly created record is consistent with the DDL specification for **rec** (either CALCed, clustered, or system-determined).

In order to create the record, a user must have write access to **rec** and to all sets in which the record type participates as an automatic owner or member. If a user does not have this security clearance, a nonzero command status is returned, the record is not created, and no currency indicator is changed.

Note 1 - If **rec** has been declared to be the AUTO member of set-1, the newly created record is automatically connected to the current owner of the set and the new record becomes the current member of the set. The connection takes place according to the member order specified in the DDL source text (such as SORTED or FIFO). If set-1 has no current owner, a nonzero command status is returned, the record is not created, and no currency indicators change.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Creation Commands

Note 2 - If **rec** has been declared to be the AUTO owner of set-2, the newly created record is automatically connected to the current member of the set and the new record becomes the current owner of the set. The connection occurs according to the set's owner order, as specified with the DDL (such as SORTED or NEXT). If set-2 has no current member, a nonzero command status is returned, the record is not created, and no currency indicators change.

Examples of Command Usage

```
block/direct   - E0 = CRS ("rec,iblk")
block/indirect - E0 = DMS ("CRS,rec,iblk")
record/direct  - E0 = CRS ("rec",iblk)
record/indirect- E0 = DMSD ("CRS,rec",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

VIII. CONNECT COMMANDS

The two DML connect commands allow you to insert existing record occurrences into a set relationship. The commands use the record that is the current of run unit and allow you to add the record as a set member or a set owner. The commands establish the relationship of the record to the set by inserting an entry for the record into the appropriate set index (as a set member or owner).

The connect commands **manually** insert a record into a set relationship; they are used in the following cases:

- the set owner or member record types have MANUAL insertion (defined in the DDL source text). When a new record is created, the data base control system does **not** connect the record to the set.

- the record needs to be connected to several owner or member record occurrences of a many-to-many (N:M) set. N:M sets allow a record to be related to many owner or member records. If the set has AUTO insertion, the specific record will be connected to one other record in the set.

- a record has been disconnected from one record of a set and needs to be reconnected to the set. Sometimes it is important to temporarily remove a particular record from a set relationship and later re-establish the set relationship.

The data base control system preserves the integrity of all set relationships. When inserting a record into a set, MDBS.DMS maintains the proper order for the set members and owners. If a user attempts to insert records that conflict with a set definition, MDBS.DMS returns a corresponding command status. Specifically, five conflicts can occur when inserting records:

- connect a member with more than one owner in a 1:1 set
- connect an owner with more than one member of a 1:1 set
- connect a member with more than one owner in a 1:N set
- connect an owner with more than one member of a N:1 set
- duplicate an existing set connection (the set member and set owner are already connected)

If a user does not have write access to the set involved in a connect command, the data base control system returns the corresponding command status; the connection does not take place.

For multiple area data bases, all areas allowing pointer indices must be on-line when performing a connect command.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Connect Commands

COMMAND DETAILS

IMS — Insert Member into Set — IMS

Command Arguments — IMS,set-1

Currency Indicators

Used	Changed	Change Description
CO(set-1) CRU	CM(set-1)	set to CRU

Description

The record that is current of run unit becomes connected to the current owner of **set-1**. This new member for the current owner of **set-1** is made the current member of **set-1**. The insertion of this new member record among the existing member records, connected to the current owner of **set-1**, is made in accordance with the member order declared for **set-1** in the DDL specification.

If the member order is NEXT (or PRIOR) then the new member is logically inserted after (or before) the set's current member; if the set has no current member, the record is logically inserted as the first (or last) member.

If multiple set owners can exist for this new member, the record that is the current owner is logically inserted into those owners according to the owner order of **set-1**. The exception is if the owner order is NEXT (or PRIOR), in which case the record that is the current owner becomes the logically first (or last) owner of the set's new current member.

The current of run unit must be an occurrence of a record type that is a member of **set-1**. A user must have write access to **set-1**, in order to connect a member record to an owner record.

Examples of Command Usage

```
block/direct   .. E0 = IMS ("set-1")
block/indirect .. E0 = DMS ("IMS,set-1")
record/direct  .. E0 = IMS ("set-1")
record/indirect.. E0 = DMS ("IMS,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Connect Commands

IOS **Insert Owner into Set** **IOS**

Command Arguments **IOS,set-1**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	set to CRU
CRU		

Description

The record that is current of run unit becomes connected to the current member of **set-1**. This new owner for the current member of **set-1** is made the current owner of **set-1**. The insertion of this new owner record among the existing owner records, connected to the current member of **set-1**, is made in accordance with the owner order declared for **set-1** in the DDL specification.

If the owner order is NEXT (or PRIOR) then the new owner is logically inserted after (or before) the set's current owner; if the set has no current owner, the record is logically inserted as the first (or last) owner.

If multiple members can exist for this new set owner, the record that is the current member is logically inserted into those members according to the member order of **set-1**. The exception is if the member order is NEXT (or PRIOR), in which case the record that is the current member becomes the logically first (or last) member of the set's new current owner.

The current of run unit must be an occurrence of a record type that is a owner of **set-1**. A user must have write access to **set-1**, in order to connect an owner record to a member record.

Examples of Command Usage

```
block/direct   .. E0 = IOS ("set-1")
block/indirect .. E0 = DMS ("IOS,set-1")
record/direct  .. E0 = IOS ("set-1")
record/indirect.. E0 = DMS ("IOS,set-1")
```

© Copyright 1987 Micro Data Base Systems, Inc.

IX. DISCONNECT COMMANDS

The MDBS DML disconnect commands allow you to remove one or more existing record occurrences from a set relationship. These commands only dissolve a single set relationship for the records involved. **They do not delete record occurrences from a data base.**

MDBS provides four disconnect commands to maintain set relationships within a data base:

- disconnect the current member record from the current set owner
- disconnect the current owner record from the current set member
- disconnect all set members from the current set owner
- disconnect all set owners from the current set member

The data base control system preserves the integrity of all set relationships. When removing one or more records from a set, MDBS.DMS maintains the proper order of the remaining set members and owners.

If a set has fixed retention, the data base control system will not disconnect any records from the set. In this case, you must delete a record from the data base to remove the set relationship for the record.

If a user does not have write access to the set involved in a disconnect command, the data base control system returns the corresponding command status. The disconnect command does not take place; the set is not altered.

For multiple area data bases, all areas allowing pointer indices must be on-line when performing a disconnect command.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Disconnect Commands

COMMAND DETAILS

RMS **Remove Member from Set** **RMS**

Command Arguments **RMS,set-1**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CM(set-1)	set to next member of set-1
CO(set-1)	CRU	set to next member of set-1

Description

Remove the connection between the current owner and current member of **set-1**. The next member (following the member that was removed) of the current owner of **set-1** becomes the new current member of **set-1** and the current of run unit. Which member is logically next depends on the member order of **set-1**, as declared in the DDL specification. If there is no next member, the current member indicator for **set-1** becomes null, the current of run unit becomes null, and the command status is 255. If a user does not have write access to **set-1**, then the current member is not disconnected, no currency indicators change, and a nonzero command status is returned.

Examples of Command Usage

```
block/direct   .. E0 = RMS ("set-1")
block/indirect .. E0 = DMS ("RMS,set-1")
record/direct  .. E0 = RMS ("set-1")
record/indirect.. E0 = DMS ("RMS,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Disconnect Commands

ROS **Remove Owner from Set** **ROS**

Command Arguments **ROS,set-1**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CO(set-1)	set to next owner of set-1
CM(set-1)	CRU	set to next owner of set-1

Description

Remove the connection between the current member and current owner of **set-1**. The next owner (following the owner that was removed) of the current member of **set-1** becomes the new current owner of **set-1** and the current of run unit. Which owner is logically next depends on the owner order of **set-1**, as declared in the DDL specification. If there is no next owner, the current owner indicator for **set-1** becomes null, the current of run unit becomes null, and the command status is 255. If a user does not have write access to **set-1**, then the current owner is not disconnected, no currency indicators change, and a nonzero command status is returned.

Examples of Command Usage

```
block/direct    .. E0 = ROS ("set-1")
block/indirect .. E0 = DMS ("ROS,set-1")
record/direct   .. E0 = ROS ("set-1")
record/indirect.. E0 = DMS ("ROS,set-1")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Disconnect Commands

RSM **Remove all Set Members** RSM

Command Arguments **RSM,set–1**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	set to null

Description

Remove the connection between the current owner of **set-1** and each one of its members. The current member of **set-1** becomes null. If a user does not have write access to **set-1**, then the members are not disconnected, the current member of **set-1** does not change, and a nonzero command status is returned.

Examples of Command Usage

```
block/direct   .. E0 = RSM ("set-1")
block/indirect .. E0 = DMS ("RSM,set-1")
record/direct  .. E0 = RSM ("set-1")
record/indirect.. E0 = DMS ("RSM,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Disconnect Commands

RSO **Remove all Set Owners** RSO

Command Arguments **RSO,set-1**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	set to null

Description

Remove the connection between the current member of **set-1** and each one of its owners. The current owner of **set-1** becomes null. If a user does not have write access to **set-1**, then the owners are not disconnected, the current owner of **set-1** does not change, and a nonzero command status is returned.

Examples of Command Usage

```
block/direct   .. E0 = RSO ("set-1")
block/indirect .. E0 = DMS ("RSO,set-1")
record/direct  .. E0 = RSO ("set-1")
record/indirect.. E0 = DMS ("RSO,set-1")
```

© Copyright 1987 Micro Data Base Systems, Inc.

X. DELETION COMMANDS

The MDBS DML delete commands eliminate record occurrences from the data base. Each delete command performs four operations to eliminate a record:

- disconnects the record from all sets connected to it
- removes the record occurrence from the data base
- returns the space occupied by the record occurrence to the data base for re-use
- updates all currency indicators referencing the deleted record

MDBS provides three delete commands to maintain the record occurrences within a data base:

- delete the current of run unit
- delete the current owner of an indicated set
- delete the current member of an indicated set

The data base control system preserves the integrity of all set relationships. When deleting a record, MDBS.DMS maintains the proper order of the remaining set members and owners for each set connected with the record.

To delete a record that is current in a user-defined currency indicator, follow these two steps:

1. make the record the current of run unit, current member of a set, or the current owner of a set
2. delete the record with the appropriate delete command

If a user does not have write access to all data items of the record type and to all sets in which that record type participates, the data base control system returns the corresponding command status. The delete command does not occur; the record and its set relationships remain intact.

For multiple area data bases, all areas allowing pointer indices must be on-line when deleting a record.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Deletion Commands

COMMAND DETAILS

DRC **Delete Record occurrence that is Current of run unit** **DRC**

Command Arguments **DRC (no arguments)**

Currency Indicators

Used	Changed	Change Description
CRU	CRU	set to null **See note 1**

Description

The record that is the current of run unit is disconnected from all set relationships and is physically deleted from the data base. The current of run unit becomes null. If a user does not have write access to the current of run unit's record type and to all sets in which that record type participates, then no deletion occurs, no currency indicator changes, and a nonzero command status is returned.

Note 1 - If the deleted record is the current owner of any set, the current member of any set, or the current record of any user-defined indicator, then those indicators become null.

Examples of Command Usage

```
block/direct   .. E0 = DRC (     )
block/indirect .. E0 = DMS ("DRC")
record/direct  .. E0 = DRC (     )
record/indirect.. E0 = DMS ("DRC")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Deletion Commands

DRM **Delete Record that is Member** **DRM**

Command Arguments **DRM,set-1**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	CM(set-1)	set to next member of set-1
CM(set-1)	CRU	set to next member of set-1 **See note 1**

Description

The record that is the current member of **set-1** is disconnected from all set relationships and is physically deleted from the data base. The next member that is owned by the current owner of **set-1** becomes the current member of **set-1** and the current of run unit. Which member is next depends on the member order of **set-1.** If there is no next member, then the current of run unit becomes null, the current member of **set-1** becomes null, and the command status is 255.

If the user does not have write access to **set-1**'s member record type and to all sets in which that record type participates, then no deletion occurs, no currency indicator changes, and a nonzero command status is returned.

Note 1 - If the record being deleted is the current owner of any set, the current member of any set other than **set-1**, or the current record of any user-defined indicator, then those indicators become null.

Examples of Command Usage

```
block/direct   .. E0 = DRM ("set-1")
block/indirect .. E0 = DMS ("DRM,set-1")
record/direct  .. E0 = DRM ("set-1")
record/indirect.. E0 = DMS ("DRM,set-1")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Deletion Commands

DRO — Delete Record that is Owner — DRO

Command Arguments DRO,set-1

Currency Indicators

Used	Changed	Change Description
CM(set-1)	CO(set-1)	set to next owner of set-1
CO(set-1)	CRU	set to next owner of set-1 **See note 1**

Description

The record that is the current owner of **set-1** is disconnected from all set relationships and is physically deleted from the data base. The next owner that is owned by the current member of **set-1** becomes the current owner of **set-1** and the current of run unit. Which owner is next depends on the owner order of **set-1**. If there is no next owner, then the current of run unit becomes null, the current owner of **set-1** becomes null, and the command status is 255.

If the user does not have write access to **set-1**'s owner record type and to all sets in which that record type participates, then no deletion occurs, no currency indicator changes, and a nonzero command status is returned.

Note 1 - If the record being deleted is the current member of any set, the current owner of any set other than **set-1**, or the current record of any user-defined indicator, then those indicators become null.

Examples of Command Usage

```
block/direct   .. E0 = DRO ("set-1")
block/indirect .. E0 = DMS ("DRO,set-1")
record/direct  .. E0 = DRO ("set-1")
record/indirect.. E0 = DMS ("DRO,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

XI. SCHEMA ACCESS COMMANDS

OVERVIEW

The MDBS Data Manipulation Language (DML) supports eight schema access commands. Each schema access command transfers structural information about the schema from the data dictionary to host language variables through a data block or program record. The commands provide structural information for data base areas, record types, data items, set relationships, set sort keys and record calc keys.

These commands do not affect any currency indicators. They only retrieve information about the structure of a specific portion of the data base.

COMMAND DETAILS

GAFN **Get Area File Name** **GAFN**

Command Arguments **GAFN,area,oblk**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The **GAFN** command retrieves the file name of a specific data base area from the data dictionary.

The **area** argument indicates the area of interest. Specify the name of a specific data base area to obtain the file name for that area.

The **oblk** contains a single variable consistent with a string data item of length 80 or 128, depending on the specific operating environment.

Examples of Command Usage

```
block/direct   .. E0 = GAFN ("area,oblk")
block/indirect .. E0 = DMS ("GAFN,area,oblk")
record/direct  .. E0 = GAFN ("area,oblk")
record/indirect.. E0 = DMSD ("GAFN",area,oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

GAI **Get Area Information** **GAI**

Command Arguments **GAI,area,oblk**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The **GAI** command retrieves information from the data dictionary about the structure of a specific data base area or all defined areas of the data base.

The **area** argument indicates the area of interest. Specify the name of a specific data base area to obtain the information for that area. Specify an asterisk (*) to obtain the information for all areas of the data base.

The **oblk** contains six variables that identify the characteristics of a data base area and are consistent with the data items in Table XI-1.

Table XI-1. Data Structure for GAI Command

Variable Position	Area Characteristic	Data Item Type	Size
First	pointers access	integer	2
Second	Page size	integer	2
Third	Number of pages	integer	2
Fourth	Area name	string	8
Fifth	Filler	char	1

The first variable returns a two byte code that includes the following information:

- pointers (sort indexes) are allowed in the area
- the last defined area in the data base

To determine pointer access for the area, perform a logical AND operation with this variable and the hexadecimal number 4. If the result is nonzero, pointers are allowed in the

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

area. Otherwise, pointers are not allowed in the area.

To determine the last defined area, perform a logical AND operation with this variable and the hexadecimal number 4000. If the result is nonzero, this is the last area returned by **GAI**. Otherwise, **GAI** returned information for another area.

The second variable returns the area page size in bytes.

The third variable returns the number of expanded pages in the area. An expanded page an existing page of the data base; it has been initialized with the default DDL initialization or the DMU expand option.

The fourth variable returns the data base area name as specified in the source DDL text.

The fifth variable is a filler variable that causes the data structure to occupy an even number of characters. No data is returned in it.

When **GAI** retrieves information for a single area, **oblk** returns the six variables. When **GAI** retrieves information for all areas, **oblk** repeats the six variables for each defined area. That is, the information of the first area is in the first six array elements; the information of the second area is in the second six array elements. Information for other areas continues in a similar manner.

When retrieving information for all areas, specify oblk as an array.

Examples of Command Usage

```
block/direct   .. E0 = GAI ("area,oblk")
block/indirect .. E0 = DMS ("GAI,area,oblk")
record/direct  .. E0 = GAI ("area,oblk")
record/indirect.. E0 = DMSD ("GAI",area,oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

GDC

Get Design Counts

GDC

Command Arguments **GDC,oblk**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The **GDC** command retrieves nine design counts from the data dictionary. Each count is a limit resulting from the DDL schema analysis and data base initialization.

The **oblk** contains ten variables that contain the design counts. The first eight variables are consistent with a 2 byte integer data item; the ninth and tenth variables are each consistent with an 18 byte character data item. The following table describes the use of each variable:

Variable position	Design Count
First	Number of areas defined for the data base
Second	Number of record types defined for the data base
Third	Number of sets defined for the data base
Fourth	maximum number of data items defined in a record type
Fifth	maximum number of record types involved in a set
Sixth	Estimated size of the largest record (data only)
Seventh	the largest number of data items in a key (sort key or calc key)
Eighth	the number of bytes in a data base key (used with SDC command)
Ninth	MDBS III version used to initiate data base (MDBS.DDL)
Tenth	filler characters; these bytes are not presently used

The first variable indicates the number of areas defined for the data base. This count includes the main data base area.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

The second variable returns the number of different record types defined in the data base. This count includes the SYSTEM record type that is part of each MDBS data base.

The third variable returns the number of sets defined in the data base. This count includes $SYSSET, a system owned set that MDBS provides within each data base definition.

The fourth variable returns the largest number of data items defined in a single record type.

The fifth variable returns the maximum number of record types involved in a set relationship. To calculate this value, each owner member record type and each member record type for the set are counted. Recursive sets (sets in which a record participates as an owner and a member) count the repeated record twice.

The sixth variable returns an estimate of the largest number of bytes the data portion of the largest could consume. This does **not** include set indexes or other internal information stored with the record.

The seventh variable returns the largest number of data items defined in a calc key or sort key.

The eighth variable returns the number of bytes allowed in a data base key. The data base key is the location of a record within the data base and can be obtained with the SDC command.

The ninth variable returns the version of MDBS III used to initiate the data base. This number is written into the data base from MDBS.DDL.

Examples of Command Usage

```
block/direct    .. E0 = GDC ("oblk")
block/indirect .. E0 = DMS ("GDC,oblk")
record/direct   .. E0 = GDC (oblk)
record/indirect.. E0 = DMSD ("GDC",oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

GHRV — Get High Range Value

GHRV

Command Arguments

GHRV,rec,itm,oblk

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The **GHRV** command retrieves the highest possible value that can exist for the data items located in record **rec**. If a range limit for the data item was specified in the DDL source text, **GHRV** returns the upper bound for the data item. Otherwise, **GHRV** returns the default upper bound for the data item.

The record **rec** identifies the record containing the data item(s) of interest.

The data item **itm** identifies the data item whose high values will be retrieved. Specify the name of a data item to retrieve a single high range value. Specify an asterisk (*) for the data item name to retrieve the high range values for all data items defined in the record.

The **oblk** variable must be consistent with the type and size of the specific data item.

When **GHRV** retrieves values for the entire record type, **oblk** is an array whose elements match the data items of the record in order, type, and size. That is, use the same data block or record structure you would use with the record retrieval commands (such as GETC).

Examples of Command Usage

```
block/direct   .. E0 = GHRV ("rec, itm, oblk")
block/indirect .. E0 = DMS ("GHRV, rec, itm, oblk")
record/direct  .. E0 = GHRV ("rec, itm",oblk)
record/indirect.. E0 = DMSD ("GHRV, rec, itm",oblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

GII **Get Item Information** **GII**

Command Arguments **GII,rec,itm,oblk**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The **GII** command retrieves the characteristics of one or all data items located in record **rec**. The characteristics include attributes such as the data item type, size, number of replications, and security access.

The record **rec** identifies the record containing the data item(s) of interest.

The data item **itm** identifies the data item whose characteristics will be retrieved. Specify the name of a data item to retrieve its characteristics. Specify an asterisk (*) for the data item name to retrieve the characteristics of all data items defined in the record.

The **oblk** contains seven variables that identify the characteristics of the data item of interest. These variables must be consistent with the data items in Table XI-2.

Table XI-2. Data Structure for GII Command

Variable position	Data Item Characteristic	Variable Type	Size
First	Size	integer	2
Second	Decimal digits	integer	2
Third	Repetitions	integer	2
Fourth	Item type	integer	2
Fifth	Item Codes	integer	2
Sixth	Item name	string	8
Seventh	Filler	char	1

The first variable returns the data item size. This is the item size specified in the DDL source text. If the data item is Internal decimal (IDEC), the data item size indicates the number of digits in the data item.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

The second variable is used only for internal decimal data items. For an IDEC items this variable returns the number of decimal digits in the item. All other data item types return zero for this variable.

The third variable returns the number of replications of a data item. The value of this variable is identical with the occurs clause specified for the data item in the DDL source text.

The fourth variable returns a number that indicates the data item type:

0 string	3 binary	6 date
1 character	4 real	7 time
2 integer	5 idec	8 unsigned

The fifth variable returns a two byte code that indicates if the data item has the following attributes. To determine whether the data item has a particular attribute, perform a logical AND operation with this variable and the value for that attribute. If the result of the logical AND operation is nonzero, the data item has the specific attribute; otherwise, the result is zero and the data item does not have the particular attribute.

10h	participates as part of a sort key
20h	has a feasibility range specified
40h	participates as part of a calc key
80h	encrypts data item occurrences
1000h	the user has read access to this data item
2000h	the user has write access to this data item
4000h	the last data item returned by GII

The sixth variable returns the name of the data item. It is identical to the name specified in the DDL source text.

The seventh variable is a filler variable that causes the data structure to occupy an even number of characters. No data is returned in it.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

When **GII** retrieves values for a single data item, **oblk** returns the six variables. When **GII** retrieves information for the entire record type, **oblk** repeats the six variables for each data item. That is, the information of the first data item is in the first six array elements. The information of the second data item is in the second six array elements; information for other data items continues in a similar manner.

When retrieving information for all data items in a record, specify oblk as an array.

Examples of Command Usage

```
block/direct   .. E0 = GII ("rec, itm, oblk")
block/indirect .. E0 = DMS ("GII, rec, itm, oblk")
record/direct  .. E0 = GII ("rec, itm",oblk)
record/indirect.. E0 = DMSD ("GII, rec, itm",oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

GKI **Get Key Information** **GKI**

Command Arguments **GKI,keystr,oblk**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The **GKI** command retrieves information about a sort key or calc key. The key information retrieved can consist of one or more data items; also, sort key information can include RECORD-TYPE.

The **keystr** argument indicates the key of interest and has one of the following forms:

"M, set-1"	retrieves sort key information for the member of set-1
"O, set-1"	retrieves sort key information for the owner of set-1
"R, rec"	retrieves calc key information for record type rec

The **oblk** contains seven variables that identify the characteristics of the data item participating as the key. These variables must be consistent with the data items in Table XI-3.

Table XI-3. Data Structure for GKI Command

Variable position	Data Item Characteristic	Variable Type	Size
First	Size	integer	2
Second	Decimal digits	integer	2
Third	Repetitions	integer	2
Fourth	Item type	integer	2
Fifth	Item Codes	integer	2
Sixth	Item name	string	8
Seventh	Filler	char	1

The first variable returns the data item size. This is the item size specified in the DDL source text. If the data item is Internal decimal (IDEC), the data item size indicates the number of digits in the data item.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

The second variable is used only for internal decimal data items. For an IDEC items this variable returns the number of decimal digits in the item. All other data item types return zero for this variable.

The third variable returns the number of replications of a data item. The value of this variable is identical with the occurs clause specified for the data item in the DDL source text.

The fourth variable returns a number that indicates the data item type:

0 string	3 binary	6 date
1 character	4 real	7 time
2 integer	5 idec	8 unsigned

The fifth variable returns a two byte code that indicates if the data item has the following attributes. To determine whether the data item has a particular attribute, perform a logical AND operation with this variable and the value for that attribute. If the result of the logical AND operation is nonzero, the data item has the specific attribute; otherwise, the result is zero and the data item does not have the particular attribute.

1h	the sort key uses descending order for this data item
2h	the sort key sorts this position by RECORD-TYPE
10h	this data item participates as part of a sort key
20h	this data item has a feasibility range specified
40h	this data item participates as part of a calc key
80h	encrypts data item occurrences
1000h	the user has read access to this data item
2000h	the user has write access to this data item
4000h	this data item is the last component of the sort key

The sixth variable returns the name of the data item. It is identical to the name specified in the DDL source text. If the set is sorted by RECORD-TYPE, this field has a null value.

The seventh variable is a filler variable that causes the data structure to occupy an even number of characters. No data it returned in it.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

When **GKI** retrieves values for a simple key (single data item), **oblk** returns the six variables. For a complex key (multiple data items), **oblk** repeats the six variables for each position of the key. That is, the information of the first position of the key, whether a data item or RECORD-TYPE, is in the first six array elements. The information of the second position of the key is in the second six array elements; information for other positions of the key continues in a similar manner.

When retrieving information for a complex key (multiple data item), specify oblk as an array.

Examples of Command Usage

```
block/direct   .. E0 = GKI ("keystr, oblk")
block/indirect .. E0 = DMS ("GKI, keystr, oblk")
record/direct  .. E0 = GKI ("keystr", oblk)
record/indirect.. E0 = DMSD ("GKI, keystr", oblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

GLRV — Get Low Range Value — GLRV

Command Arguments **GLRV,rec,itm,oblk**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The **GLRV** command retrieves the lowest possible value that can exist for the data items located in record **rec**. If a range limit for the data item was specified in the DDL source text, **GLRV** returns the lower bound for the data item. Otherwise, **GLRV** returns the default lower bound for the data item.

The record **rec** identifies the record containing the data item(s) of interest.

The data item **itm** identifies the data item whose low values will be retrieved. Specify the name of a data item to retrieve a single low range value. Specify an asterisk (*) for the data item name to retrieve the low range values for all data items defined in the record.

The **oblk** variable must be consistent with the type and size of the specific data item.

When **GHRV** retrieves values for the entire record type, **oblk** is an array whose elements match the data items of the record in order, type, and size. That is, use the same data block or record structure you would use with the record retrieval commands (such as GETC).

Examples of Command Usage

```
block/direct   .. E0 = GLRV ("rec, itm, oblk")
block/indirect .. E0 = DMS ("GLRV, rec, itm, oblk")
record/direct  .. E0 = GLRV ("rec, itm", oblk)
record/indirect.. E0 = DMSD ("GLRV, rec, itm", oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

GRI **Get Record Information** **GRI**

Command Arguments **GRI,recstr,oblk**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The **GRI** command retrieves structural information about one or more record types from the data base.

The **recstr** identifies the records of interest and has one of the following forms:

"I, itm"	retrieves information of every record type that contains a data item with the name itm
"M, set-1"	retrieves information of every record type that is a member of set-1
"O, set-1"	retrieves information of every record type that is an owner of set-1
"R, rec"	retrieves information of record type rec
"S, set-1"	retrieves information of every record type that participates (both owner and member) in set-1
"*"	retrieves information of every record type in the data base schema

The **oblk** contains five variables that identify the characteristics of the record. These variables must be consistent with the data items of table XI-4.

Table XI-4. Data Structure for GRI Command

Variable position	Record Characteristic	Data Item Type	Size
First	Record Codes	integer	2
Second	Size	integer	2
Third	Data item count	integer	2
Fourth	Record name	string	8
Fifth	Clustering set name	string	8

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

The first variable returns a two byte code that indicates if the record has the following attributes. To determine whether the record type has a particular attribute, perform a logical AND operation with this variable and the value for the attribute. If the result of the logical AND operation is nonzero, the record type has the specific attribute; otherwise, the result is zero and the record type does not have the particular attribute.

1h	the record type has a calc key
2h	the record type allows duplicate calc keys for record occurrences
4h	this record type clusters record occurrences in the area of the owner record type of a particular set (variable 5 contains set name)
8h	this record type clusters record occurrences in the area of the member record type of a particular set (variable 5 contains set name)
1000h	the user has read access to this record type
2000h	the user has write access to this record type
4000h	the last record type returned by GRI

The second variable returns an estimate of the largest number of bytes the data portion of this record could occupy. This does **not** include set indices or other internal information stored with the record.

The third variable returns the number of data items defined in the record.

The fourth variable returns the name of the record type. It is identical to the name specified in the DDL source text.

The fifth variable returns the name of the set used for clustering this record. If the record type does not use clustering, this variable has a null value.

When **GRI** retrieves values for a single record type, **oblk** returns the five variables. When **GII** retrieves information for several record types, **oblk** repeats the five variables for each data item. That is, the information of the first record type is in the first five array elements. The information of the second record type is in the second five array elements; information for other record types continues in a similar manner.

When retrieving information for several record types, specify oblk as an array.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

Examples of Command Usage

block/direct .. E0 = GRI ("recstr, oblk")
block/indirect .. E0 = DMS ("GRI, recstr, oblk")
record/direct .. E0 = GRI ("recstr", oblk)
record/indirect.. E0 = DMS ("GRI, recstr",oblk)

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

GSI **Get Set Information** **GSI**

Command Arguments **GSI,setstr,oblk**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The **GSI** command retrieves structural information about one or more set relationships from the data base.

The **setstr** identifies the sets of interest and has one of the following forms:

"R, rec"	retrieves information of every set in which record type rec participates
"S, set-1"	retrieves information of set set-1
"*"	retrieves information of every set in the data base schema

The **oblk** contains eight variables that identify the characteristics of a set. These variables must be consistent with the data items of table XI-5.

Table XI-5. Data Structure for GSI Command

Variable position	Set Characteristic	Variable Type	Size
First	Type	integer	2
Second	owner order	integer	2
Third	owner count	integer	2
Fourth	member order	integer	2
Fifth	member count	integer	2
Sixth	Set codes	integer	2
Seventh	Set name	string	8
Eighth	Filler	char	3

The first variable returns a number that indicates the type of set relationship:

0	1:1 relationship	2	1:N relationship
1	N:1 relationship	4	N:M relationship

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

The second variable returns a number that indicates the order for owner record occurrences:

0 Sorted	2 Last In First Out	4 Next
1 Immaterial	3 First In First Out	5 Prior

The third variable returns the number of owner record types defined for the set.

The fourth variable returns a number that indicates the order for member record occurrences. The values of this variable are identical to those used for the owner record order (second variable).

The fifth variable returns the number of member record types defined for the set.

The sixth variable returns a two byte code that indicates if the set has the following attributes. To determine whether the set has a particular attribute, perform a logical AND operation with this variable and the value for that attribute. If the result of the logical AND operation is nonzero, the set has the specific attribute; otherwise, the result is zero and the set does not have the particular attribute.

1h	the set members are inserted automatically by the data base control system when the records are created
2h	For sets with sorted member record occurrences, duplicate member sort keys are ordered Immaterial
4h	For sets with sorted member record occurrences, duplicate member sort keys are ordered First In First Out (LIFO)
8h	For sets with sorted member record occurrences, duplicate member sort keys are ordered Last In First Out (FIFO)
10h	the set owners are inserted automatically by the data base control system when the records are created
20h	For sets with sorted owner record occurrences, duplicate owner sort keys are ordered Immaterial
40h	For sets with sorted owner record occurrences, duplicate owner sort keys are ordered First In First Out (LIFO)
80h	For sets with sorted owner record occurrences, duplicate owner sort keys are ordered Last In First Out (FIFO)

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Schema Access Commands

100h	the set has fixed retention
1000h	the user has read access to this set
2000h	the user has write access to this set
4000h	the last set returned by GSI

The seventh variable returns the name of the set. It is identical to the name specified in the DDL source text.

The eighth variable is a filler variable whose contents are reserved for future use. This variable also causes the data structure to occupy an even number of characters. No data is returned in it.

When **GSI** retrieves values for a single set, **oblk** returns the seven variables. When **GSI** retrieves information for several sets, **oblk** repeats the seven variables for each set. That is, the information of the first set is in the first seven array elements. The information of the second set is in the second seven array elements; information for sets continues in a similar manner.

When retrieving information for several sets, specify oblk as an array.

Examples of Command Usage

```
block/direct    .. E0 = GSI ("setstr, oblk")
block/indirect .. E0 = DMS ("GSI, setstr, oblk")
record/direct   .. E0 = GSI ("setstr", oblk)
record/indirect.. E0 = DMSD ("GSI, setstr", oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

XII. UTILITY COMMANDS

MDBS III provides twenty-three utility commands. The commands address nine miscellaneous areas not covered by other DML commands:

- Allocate user-defined currency indicators
  - AUI

- Clear (Null) all currency indicators
  - NCI

- Customize the data base control system's behavior
  - DBCNV, DBENV

- Flush the page buffers of the data base control system
  - DBSAVE

- Open and close a data base or data base area
  - DBOPN, DBOPNA
  - DBCLS, DBCLSA

- Retrieve the number of set owners (members) for the current set member (owner)
  - GMC, GOC

- Retrieve the record type for the current of run unit, the current set member, or the current set owner
  - GTC, GTM, GTO

- Retrieve usage statistics for the data base
  - DBSTAT

- Test the contents of currency indicators for equality or a specific record type
  - CCU
  - TCT, TMT, TOT

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

COMMAND DETAILS

AUI **Allocate User Indicators** **AUI**

Command Arguments **AUI,iblk**

Currency Indicators

Used	Changed	Change Description
none	none;	**See note 1**

Description

One host language variable is defined for **iblk.** It must be consistent with a one byte unsigned data item. The value of that variable must be between 0 and 255, inclusive. This value tells MDBS.DMS how many user-defined indicators are to be allocated. If the value is 0, then no user-defined indicators are allocated. If the value is 139, the user indicators 1 through 139 are allocated. All user defined indicators that have been allocated are available for use by the application program and can reference any record occurrence.

AUI can be invoked repeatedly within an application program to increase or decrease the number of user-defined indicators, as desired. When a data base is opened for data manipulation (with the DBOPN command), four user-defined indicators are automatically allocated and set to null. If an application program does not use a Boolean command and does not need any user indicator, the AUI can be used to allocate zero user indicators; this conserves space in the page buffer region. If more than four user indicators are needed, use AUI to allocate the desired number.

When AUI is employed to increase the number of user indicators, the records that were current for already allocated user indicators are still current. The new indicators are set to null. When AUI is employed to decrease the number of user indicators, only those records that are current for the resultant user indicators are still current. For instance, if ten user indicators are allocated and AUI is used to reduce this to five, then only those records that were current for user indicators 1 through 5 are still current after invoking AUI.

The first two user indicators (1,2) are needed when performing a Boolean DML command.

Note 1 -When AUI decreases the number of available user indicators, the data base control system does not maintain knowledge about which records were current for user indicators that no longer exist.

Examples of Command Usage

```
block/direct   .. E0 = AUI ("iblk")
block/indirect .. E0 = DMS ("AUI,iblk")
record/direct  .. E0 = AUI (iblk)
record/indirect.. E0 = DMSD ("AUI",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

CCU — Check Current of run unit against User indicator — CCU

Command Arguments **CCU,iblk**

Currency Indicators

Used	Changed	Change Description
CRU	none	
CU(i)		

Description

Check to determine whether the record that is current for a user-defined indicator is the same record that is the current of run unit. The value of **iblk**'s host language variable determines the user indicator for which this check is made. That variable must be consistent with a one byte, unsigned data item. If the current of run unit is the same record as the specified user indicator, then the command status is 0. If it is not, then the command status is 255.

Examples of Command Usage

```
block/direct   .. E0 = CCU ("iblk")
block/indirect .. E0 = DMS ("CCU,iblk")
record/direct  .. E0 = CCU (iblk)
record/indirect.. E0 = DMSD ("CCU",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

DBCLS

Data Base CLoSe

DBCLS

Command Arguments

DBCLS (no arguments)

Currency Indicators

Used	Changed	Change Description
none	all	set to null

Description

In a single-user environment, DBCLS nulls all of that program's currency indicators, eliminates all of the user indicators for the program, flushes the page buffers (that is, rewrites all changed pages from main memory into auxiliary memory), and performs a TRCOM (for the RTL form of the system). No further data manipulation can be performed after DBCLS until the data base is reopened (with DBOPN).

Failure to close a data base can leave its contents inconsistent, if the data base was opened for modification. This inconsistency will prevent subsequent efforts to open the data base. The data base can be restored to a consistent state with the MDBS RCV utility.

In a multi-user environment, many application programs share the same page buffers. Invoking DBCLS within an application program nulls out all the currency indicators and eliminates all of the user indicators of that program. It also flushes the page buffers (after a specific amount of idle time) if no other application program currently has the data base open for processing. In order to force the page buffers to flush, regardless of other run units, use the DBSAVE command. An application program can perform no further data manipulation after it has invoked DBCLS until it reopens the data base (with DBOPN).

Note - DBCLS should be the last DML command invoked in an application program in both the single user and multiuser environments.

Examples and Command Usage

```
block/direct   .. E0 = DBCLS (     )
block/indirect .. E0 = DMS ("DBCLS")
record/direct  .. E0 = DBCLS (     )
record/indirect.. E0 = DMS ("DBCLS")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

DBCLSA **Data Base CLoSe Area** **DBCLSA**

Command Arguments **DBCLSA,area**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

DBCLSA flushes all page buffers that contain pages from the indicated **area**. Flush means to rewrite changed pages from main memory into auxiliary memory. The file on which the area resides is closed by the operating system.

Note - DBCLSA is available only in single user environments.

Examples of Command Usage

```
block/direct   .. E0 = DBCLSA ("area")
block/indirect .. E0 = DMS ("DBCLSA,area")
record/direct  .. E0 = DBCLSA ("area")
record/indirect.. E0 = DMS ("DBCLSA,area")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

DBCNV — Data Base format CoNVersion — DBCNV

Command Arguments **DBCNV,iblk**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The DBCNV command controls the format of date data items that are transferred between the data base control system and application programs. Also, DBCNV provides control over the input of null dates and times, whether or not the data base control system allows them as valid data values.

iblk contains one host language variable that is consistent with a one byte, unsigned data item. The acceptable data values for iblk are one through eight.

The default format for date data items is mm/dd/yyyy. To change from one date format to another, specify **iblk** to contain a value of one to four as indicated below:

1 mm/dd/yyyy (the standard MDBS date format)
2 dd/mm/yyyy
3 yyyy/mm/dd
4 yyyy/dd/mm

The data base control system automatically converts date values from an internal form to the specified format as they are transferred from the data base to that run unit. As a value is transferred to a data base, MDBS.DMS converts the the date from the input format to the internal form.

DBCNV can be invoked multiple times within a run unit to switch between various date formats, but the data base control system has only one format in effect at a time.

DBCNV can also be used to allow (or disallow) null date or time values. The standard MDBS default condition is that null dates and null times are not allowed. Permissible **iblk** variable values and their effects are as follows:

5 null dates are allowed on input
6 null dates are not allowed on input (default condition)

7 null times are allowed on input
8 null times are not allowed on input (default condition)

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

Here, "input" refers to any attempt to create or alter the value of a date or time data item. If a null date or time is involved in a sorted relationship, null values are sequenced after non- null values.

Examples of Command Usage

```
block/direct    .. E0 = DBCNV ("iblk")
block/indirect  .. E0 = DMS ("DBCNV,iblk")
record/direct   .. E0 = DBCNV (iblk)
record/indirect .. E0 = DMSD ("DBCNV",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

DBENV **Data Base ENVironment options** **DBENV**

Command Arguments **DBENV,iblk**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

A run unit can use this command to customize its processing environment by specifying how the data base control system should behave when interacting with that run unit. The value of **iblk**'s host language variable governs which aspect of the environment is altered. If the value is 1, 2, or 3 then MDBS.DMS behavior is altered for all run units for as long as any of them still has the data base open. The host language variable for **iblk** must be consistent with a one byte unsigned data item. Since option 1 alleviates some of the checking that is otherwise performed by MDBS.DMS, it can increase the processing speed of an application program that modifies a data base. This option should only be used in an application program that is known to be "bug" free.

If the variable has a value of 1, then MDBS.DMS does not perform a check to determine whether a record being connected to an owner record for a set is already owned by that owner record via that set.

If the variable has a value of 2, then checksums on disk reads are not suppressed. This option provides additional integrity measures, but results in slower processing.

If the variable has a value of 3, the MDBS.DMS automatically performs a DBSAVE whenever a DML command causes a change to a page(s) in memory. As with option two, this option provides additional integrity safeguards, but results in slower processing.

If iblk's variable has a value of 4, then no passive lock checking will be performed for the run unit that invokes DBENV. Though this option results in faster processing of the run unit's DML commands, it should be used only with extreme caution because it eliminates a significant portion of the built-in integrity checking. When its passive lock checking is disabled, the run unit will be allowed to alter records that are passively locked by other run units. It will not be allowed to actively lock a record that is passively locked by others. The usual record deletion and set removal integrity restrictions are still enforced (see Table XV-1).

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

If iblk's variable has a value of 5, the run unit's attempts to find a record based on a sort key (e.g., with FMSK) will be processed in a slightly different way than usual. If a record with an indicated sort key value cannot be found (because it does not exist), then the pertinent currency indicators (CRU and either CO or CM) become null and the command status is 255.

If the variable has a value of 6, the automatic FFM or FFO that is normally carried out for assignment commands (e.g., SOM) is not performed. In some situations this allows the run unit's control logic to be more concise.

After DBENV has been invoked to alter some aspect of the environment, that aspect of the environment can be restored to its default behavior. This is accomplished by invoking DBENV again, using a variable value that is 100 greater than what was previously used. For instance, if **iblk**'s variable has a value of 105, then the effect of formerly invoking DBENV with a value of 5 is no longer in force. If DBENV is invoked with a value of 101, 102, or 103 then the normal MDBS.DMS behavior is restored for all run units.

Examples of Command Usage

```
block/direct   .. E0 = DBENV ("iblk")
block/indirect .. E0 = DMS ("DBENV,iblk")
record/direct  .. E0 = DBENV (iblk)
record/indirect.. E0 = DMSD ("DBENV",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

DBOPN — Data Base OPeN — DBOPN

Command Arguments **DBOPN,iblk**

Currency Indicators

Used	Changed	Change Description
none	CRU	set to system record
	CO(system-owned sets)	set to system record
	CO(other sets)	set to null
	CU(1)	set to null
	CU(2)	set to null
	CU(3)	set to null
	CU(4)	set to null

Description

DBOPN opens a data base for processing by an application program. An application program must invoke the DBOPN command before any data held in the data base can be accessed. There are four host language variables for the **iblk** data block (or program record type), which MDBS.DMS expects in the following sequence:

The first variable contains a user name that has been specified in the DDL. This variable must be consistent with a string data item of no more than 16 bytes.

The second variable contains the password for the user as specified in the DDL. This variable must be consistent with a string data item of no more than 12 bytes.

The third variable indicates the type of processing that the program will perform. This variable must be consistent with a string data item of no more than 4 bytes. If the variable's value begins with R, the program is allowed to perform retrieval only (that is, read access only, regardless of a user's write access codes). If it begins with M, the program can modify (as well as retrieve) data. If the variable's value begins with E, the run unit has exclusive modification and retrieval access to a data base, subject to security constraints;

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

in a multiuser environment this prohibits any other application program from accessing a data base, until the program with exclusive access invokes DBCLS. Since it is not sharing the data base with other run units, a run unit with exclusive access will execute faster than if it had R or M access.

The fourth variable contains the fully qualified file name for the main **area** of the data base being opened. This variable must be consistent with a string data item; the maximum length is operating system dependent. It cannot exceed the length of a fully qualified file name permitted in the operating system (consult the appropriate system specific manual).

In block-oriented languages, the **iblk** data block must be the last data block defined (see DEFINE command) before invoking DBOPN. This block and all other data blocks defined before invoking DBOPN are inoperative after invoking DBOPN. (They must be defined again, after DBOPN, in order to be used after DBOPN.)

Upon opening a data base for processing the current of run unit and the current owners of all system-owned sets become the SYSTEM record occurrence. All other currency indicators are null. Four user-defined currency indicators are automatically allocated.

The page-image file must have been declared with the PIFD command (Chapter XVI) prior to invoking DBOPN. For the RTL form, a command status warning of 74 is returned if no log file is present; this does not prevent the data base from being opened.

In a single user environment, DBOPN opens only the main **area** of the data base; other **areas** are automatically opened by MDBS.DMS when the application program needs to access records in those **areas**. These are opened for the same type of processing as the main **area**. A nonzero command status results if an **area** that needs to be automatically opened is not on-line.

In a multiuser environment, the multiuser data base control system opens all the data base areas before any application run unit begins.

Examples and Command Usage

```
block/direct   .. E0 = DBOPN ("iblk")
block/indirect .. E0 = CALL DMS ("DBOPN,iblk")
record/direct  .. E0 = DBOPN (iblk)
record/indirect.. E0 = CALL DMSD ("DBOPN",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

DBOPNA **Data Base OPeN Area** **DBOPNA**

Command Arguments

DBOPNA,area

Currency Indicators

Used	Changed	Change Description
none	none	

Description

DBOPNA explicitly opens the indicated **area** for processing. MDBS.DMS expects to find this **area** on the file indicated by the value of **iblk**'s host language variable. This variable must be consistent with a string data item; its size is operating system dependent, but cannot exceed the maximum length of a fully qualified file name allowed by the host operating system (consult the pertinent system specific manual). The value of the variable must be the fully qualified name of a file that is on-line when DBOPNA is invoked. This name does not need to match the file name assigned to the **area** in the DDL specifications. If the indicated file is not on-line, a command status of 54 is returned. This fact can be used by the application developer to determine whether an end user of the application program has the appropriate disks on-line; if they are not, a corresponding prompt can be issued to the end user.

If **iblk**'s host language variable has a null value, then MDBS.DMS assumes that the **area** exists on the file specified for that **area** in the DDL specifications (or its default name, if none is specified).

Note - DBOPNA is available only in single user environments.

Examples and Command Usage

```
block/direct  .. E0=DBOPNA("area,iblk")          or E0 = DBOPNA ("area")
block/indirect .. E0 = DMS ("DBOPNA,area,iblk")  or E0 = DMS ("DBOPNA,area")
record/direct  .. E0 = DBOPNA ("area",iblk)
record/indirect.. E0 = CALL DMSD ("DBOPNA,area",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

DBSAVE

Data Base SAVE

DBSAVE

Command Arguments **DBSAVE (no arguments)**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

DBSAVE flushes the page buffers, regardless of whether the run unit is executing in a single-user or multiuser environment. This saves all changes that have been made to pages (in the page buffers) since they entered main memory. These changed pages are saved by rewriting them into the data base that exists in auxiliary memory.

It is advisable to invoke DBSAVE after extremely crucial transactions that change data base contents. This guarantees that those transactions are immediately incorporated into the permanent copy of a data base. In the event of an abnormal interruption of a run unit, the following rule holds. If no changes have been made to the data base since the last DBSAVE, then the data base is consistent and can be re-opened successfully (i.e., without a command status of 15).

In a multiuser environment, the data base control system may periodically invoke DBSAVE automatically, independent of any executing run unit. This occurs when the data base control system is idle for a short time and protects the data base by making it consistent. Thus an abnormal interruption may still leave the data base in a consistent state so it can be re-opened successfully.

Examples of Command Usage

```
block/direct   .. E0 = DBSAVE (     )
block/indirect .. E0 = DMS ("DBSAVE")
record/direct  .. E0 = DBSAVE (     )
record/indirect.. E0 = DMS ("DBSAVE")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

DBSTAT — Data Base STATistics — DBSTAT

Command Arguments

DBSTAT,oblk

Currency Indicators

Used	Changed	Change Description
none	none	

Description

Statistics on data base utilization are returned as values of **oblk**'s five host language variables. Each of these variables must be consistent with a two byte, unsigned data item. The sequence of the five returned statistics is as follows:

1. the number of page buffers currently allocated in main memory,

2. the number of times (since opening the data base) that the most recent page access was to a different page than the last page access,

3. the number of read requests issued by MDBS.DMS since opening the data base,

4. the number of write requests issued since the data base was opened that were due to DBSAVE or to background processing in a multiuser environment,

5. the total number of write requests issued by MDBS.DMS since opening the data base.

Examples of Command Usage

```
block/direct   .. E0 = DBSTAT ("oblk")
block/indirect .. E0 = DMS ("DBSTAT,oblk")
record/direct  .. E0 = DBSTAT (oblk)
record/indirect.. E0 = DMSD ("DBSTAT",oblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

GMC — Get Member Count — GMC

Command Arguments **GMC,set−1,oblk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	none	

Description

The number of member records connected to the current owner of **set-1** is returned in the host language variable defined for **oblk**. This variable must be consistent with a four byte, unsigned data item (see system specific manuals). If a user does not have read access to **set-1**, the member count is not returned and a nonzero command status is issued.

Examples of Command Usage

```
block/direct   .. E0 = GMC ("set-1,oblk")
block/indirect .. E0 = DMS ("GMC,set-1,oblk")
record/direct  .. E0 = GMC ("set-1",oblk)
record/indirect.. E0 = DMSD ("GMC,set-1",oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

GOC — Get Owner Count — GOC

Command Arguments **GOC,set−1,oblk**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	none	

Description

The number of owner records connected to the current member of **set-1** is returned in the host language variable defined for **oblk**. This variable must be consistent with a four byte, unsigned data item (see system specific manuals). If a user does not have read access to **set-1**, the owner count is not returned and a nonzero command status is issued.

Examples of Command Usage

```
block/direct   .. E0 = GOC ("set-1,oblk")
block/indirect .. E0 = DMS ("GOC,set-1,oblk")
record/direct  .. E0 = GOC ("set-1",oblk)
record/indirect.. E0 = DMSD ("GOC,set-1",oblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

GTC — Get Type of Current of run unit — GTC

Command Arguments

GTC,oblk

Currency Indicators

Used	Changed	Change Description
CRU	none	

Description

The name of the record type for the record that is current of run unit is returned in **oblk**'s host language variable. That variable must be consistent with an eight byte, character data item.

Examples of Command Usage

```
block/direct   .. E0 = GTC ("oblk")
block/indirect .. E0 = DMS ("GTC,oblk")
record/direct  .. E0 = GTC (oblk)
record/indirect.. E0 = DMSD ("GTC",oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

GTM **Get Type of Member** GTM

Command Arguments **GTM,set-1,oblk**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	none	

Description

The name of the record type for the record that is the current member of **set-1** is returned in **oblk**'s host language variable. That variable must be consistent with an eight byte, character data item. If a user does not have read access to **set-1**, a nonzero command status is returned and the value of **oblk**'s host language variable is unchanged.

Examples of Command Usage

```
block/direct   .. E0 = GTM ("set-1,oblk")
block/indirect .. E0 = DMS ("GTM,set-1,oblk")
record/direct  .. E0 = GTM ("set-1",oblk)
record/indirect.. E0 = DMSD ("GTM,set-1",oblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

GTO — Get Type of Owner — GTO

Command Arguments **GTO,set-1,oblk**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	none	

Description

The name of the record type for the record that is the current owner of **set-1** is returned in **oblk**'s host language variable. That variable must be consistent with an eight byte, character data item. If a user does not have read access to **set-1**, a nonzero command status is returned and the value of **oblk**'s host language variable is unchanged.

Examples of Command Usage

```
block/direct   .. E0 = GTO ("set-1,oblk")
block/indirect .. E0 = DMS ("GTO,set-1,oblk")
record/direct  .. E0 = GTO ("set-1",oblk)
record/indirect.. E0 = DMSD ("GTO,set-1"oblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

NCI — Null all Currency Indicators — NCI

Command Arguments — **NCI (no arguments)**

Currency Indicators

Used	Changed	Change Description
none	All	set to null

Description

All currency indicators are made null, with the exceptions of current owners of system-owned sets. The active lock statuses are not affected. Thus, if MCP is in effect, NCI nulls the current of run unit; but as soon as the current of run unit again becomes non-null, its record is actively locked.

Examples of Command Usage

```
block/direct   .. E0 = NCI (     )
block/indirect .. E0 = DMS ("NCI")
record/direct  .. E0 = NCI (     )
record/indirect.. E0 = DMS ("NCI")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

TCN — Test Current of run unit for Null — TCN

Command Arguments

TCN (no arguments)

Currency Indicators

Used	Changed	Change Description
CRU	none	

Description

A command status of 255 is returned if the current of run unit is null. If it is not null, the command status is 0.

Examples of Command Usage

```
block/direct    .. E0 = TCN (     )
block/indirect  .. E0 = DMS ("TCN")
record/direct   .. E0 = TCN (     )
record/indirect .. E0 = DMS ("TCN")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

TCT **Test Current of run unit Type** TCT

Command Arguments **TCT,rec**

Currency Indicators

Used	Changed	Change Description
CRU	none	

Description

The record type of the current of run unit is compared to **rec**. If they are the same, the command status is 0. If they are not the same, the command status is 3.

Examples of Command Usage

```
block/direct   .. E0 = TCT ("rec")
block/indirect .. E0 = DMS ("TCT,rec")
record/direct  .. E0 = TCT ("rec")
record/indirect.. E0 = DMS ("TCT,rec")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

TMN **Test Member for Null** TMN

Command Arguments **TMN,set-1**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	none	

Description

If the current member of **set-1** is null, a command status of 255 is returned. If it is not null, the command status is 0.

Examples of Command Usage

```
block/direct    .. E0 = TMN ("set-1")
block/indirect  .. E0 = DMS ("TMN,set-1")
record/direct   .. E0 = TMN ("set-1")
record/indirect .. E0 = DMS ("TMN,set-1")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

TMT **Test Member Type** **TMT**

Command Arguments **TMT,rec,set-1**

Currency Indicators

Used	Changed	Change Description
CM(set-1)	none	

Description

The record type of the current member of **set-1** is compared to **rec**. If they are the same, the command status is 0. If they are not the same, the command status is 3.

Examples of Command Usage

```
block/direct   .. E0 = TMT ("rec,set-1")
block/indirect .. E0 = DMS ("TMT,rec,set-1")
record/direct  .. E0 = TMT ("rec,set-1")
record/indirect.. E0 = DMS ("TMT,rec,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

TON — Test Owner for Null — TON

Command Arguments TON,set-1

Currency Indicators

Used	Changed	Change Description
CO(set-1)	none	

Description

If the current owner of **set-1** is null, a command status of 255 is returned. If it is not null, the command status is 0.

Examples of Command Usage

```
block/direct    .. E0 = TON ("set-1")
block/indirect  .. E0 = DMS ("TON,set-1")
record/direct   .. E0 = TON ("set-1")
record/indirect .. E0 = DMS ("TON,set-1")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

TOT **Test Owner Type** TOT

Command Arguments **TOT,rec,set-1**

Currency Indicators

Used	Changed	Change Description
CO(set-1)	none	

Description

The record type of the current owner of **set-1** is compared to **rec**. If they are the same, the command status is 0. If they are not the same, the command status is 3.

Examples of Command Usage

```
block/direct    .. E0 = TOT ("rec,set-1")
block/indirect .. E0 = DMS ("TOT,rec,set-1")
record/direct  .. E0 = TOT ("rec,set-1")
record/indirect.. E0 = DMS ("TOT,rec,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Utility Commands

TUN **Test User indicator for Null** **TUN**

Command Arguments **TUN,iblk**

Currency Indicators

Used	Changed	Change Description
CU(i)	none	

Description

The value of **iblk**'s host language variable determines the user indicator which is tested for being null. That variable must be consistent with a one byte, unsigned data item. A 255 command status is returned if the indicated user indicator is null. If it is not null, then a 0 command status is returned.

Examples of Command Usage

```
block/direct   .. E0 = TUN ("iblk")
block/indirect .. E0 = DMS ("TUN,iblk")
record/direct  .. E0 = TUN (iblk)
record/indirect.. E0 = DMS ("TUN",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

XIII. BOOLEAN COMMANDS

The MDBS Data Manipulation Language provides eight commands that perform Boolean operations and produce a group of records satisfying the Boolean condition. The DML Boolean commands perform two operations:

- Boolean Intersection (logical AND)
  The Boolean intersect commands use two groups of record occurrences and find the record occurrences common to both groups. The data base control system connects these common record occurrences as set members to a system-owned set.

  - AND set members of record group one with set members of record group two
  - AND set members of record group one with set owners of record group two
  - AND set owners of record group one with set members of record group two
  - AND set owners of record group one with set owners of record group two

- Boolean Difference (logical difference)
  The Boolean difference commands use two groups of record occurrences and find the record occurrences exclusive to the first record group. The data base control system connects the exclusive record occurrences as set members to a system-owned set.

  - Exclude set members of record group two from set members of record group one
  - Exclude set members of record group two from set owners of record group one
  - Exclude set owners of record group two from set members of record group one
  - Exclude set owners of record group two from set owners of record group one

Each record group for the DML Boolean commands consists of the set members or set owners of a particular record occurrence. For example, suppose a data base consists of the following schema:

Record DEPT - department record

Record EMP - employee record

Set DEPTS - system owned set to all departments

Set CONTAINS - Set linking each department to all of its employees

The employees of any department can be a record group, by identifying the particular department record and by specifying the members of the set CONTAIN in the DML intersect or difference command.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Boolean Commands

To perform a DML Boolean command, follow these steps:

1. Select record group one
   - Find the record (set owner or member) that refers to the appropriate record group
   - Set user-defined indicator one to this record

2. Select record group two
   - Find the record (set owner or member) that refers to the appropriate record group
   - Set user-defined indicator two to this record

3. Specify the proper DML Boolean command, matching the Boolean operation and record groups with the command mnemonic

4. Specify the proper command arguments:
   - argument one names the set of record group one
   - argument two names the set of record group two
   - argument three names the set containing the resultant record occurrences

Each Boolean command uses the records assigned to the first two user-defined indicators. These currency indicators must be allocated prior to invoking the Boolean command. The data base open command (DBOPN) automatically allocates the first four user-defined indicators when it opens the data base and the allocate user indicator command (AUI) re-allocates them if necessary.

The input sets can be identical and can refer to the same record group, if desired. The system owned set $SYSSET is also available as an input set.

The resultant set (the last command argument) is a system-owned set (such as $SYSSET) whose member record occurrences allow all record types involved in the Boolean operation. The resultant set can be identical to one of the input sets, however, the contents of the set will change completely. MDBS.DMS removes all set members from the resultant set before connecting records from the DML Boolean command.

For multiple area data bases, all areas allowing pointer indices must be on-line when performing a Boolean DML command.

Note: The user of a Boolean command must have **read access** to the input sets and **write access** to the resultant set.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Boolean Commands

COMMAND DETAILS

AMM — And of Members with Members — AMM

Command Arguments **AMM,set−1,set−2,set−3**

Currency Indicators

Used	Changed	Change Description
CU(1)	CM(set-3)	set to null
CU(2)	CRU	set to null

Description

Perform a Boolean intersection on record groups one and two and place the records common to both record groups in **set-3** as members.

Record group one consists of the **member** record occurrences of **set-1** connected to the record assigned to user-defined indicator one.

Record group two consists of the **member** record occurrences of **set-2** connected to the record assigned to user-defined indicator two.

Any member of **set-3** that is not in the intersection is disconnected from **set-3** during the execution of AMM.

If the member order of **set-3** is not sorted, then the set member order of **set-3** after AMM is a function of the order of record groups one and two.

It is not reasonable to use AMM unless record groups one and two have at least one record type in common. Otherwise, the intersection will be empty and **set-3** will have no members. **If set-3 is not $SYSSET, then it must allow each possible member record type for record groups one and two.**

The command status becomes 255 after successfully running AMM.

Examples of Command Usage

```
block/direct   .. E0 = AMM ("set-1,set-2,set-3")
block/indirect .. E0 = DMS ("AMM,set-1,set-2,set-3")
record/direct  .. E0 = AMM ("set-1,set-2,set-3")
record/indirect.. E0 = DMS ("AMM,set-1,set-2,set-3")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Boolean Commands

AMO — And of Members with Owners

AMO

Command Arguments **AMO,set−1,set−2,set−3**

Currency Indicators

Used	Changed	Change Description
CU(1)	CM(set-3)	set to null
CU(2)	CRU	set to null

Description

Perform a Boolean intersection on record groups one and two and place the records common to both record groups in **set-3** as members.

Record group one consists of the **member** record occurrences of **set-1** connected to the record assigned to user-defined indicator one.

Record group two consists of the **owner** record occurrences of **set-2** connected to the record assigned to user-defined indicator two.

Any member of **set-3** that is not in the intersection is disconnected from **set-3** during the execution of AMO.

If the member order of **set-3** is not sorted, then the set member order of **set-3** after AMO is a function of the order of record groups one and two.

It is not reasonable to use AMO unless record groups one and two have at least one record type in common. Otherwise, the intersection will be empty and **set-3** will have no members. **If set-3 is not $SYSSET, then it must allow each possible member record type for record groups one and two.**

The command status becomes 255 after successfully running AMO.

Examples of Command Usage

```
block/direct   .. E0 = AMO ("set-1,set-2,set-3")
block/indirect .. E0 = DMS ("AMO,set-1,set-2,set-3")
record/direct  .. E0 = AMO ("set-1,set-2,set-3")
record/indirect.. E0 = DMS ("AMO,set-1,set-2,set-3")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Boolean Commands

AOM — And of Owners with Members — AOM

Command Arguments **AOM,set−1,set−2,set−3**

Currency Indicators

Used	Changed	Change Description
CU(1)	CM(set-3)	set to null
CU(2)	CRU	set to null

Description

Perform a Boolean intersection on record groups one and two and place the records common to both record groups in **set-3** as members.

Record group one consists of the **owner** record occurrences of **set-1** connected to the record assigned to user-defined indicator one.

Record group two consists of the **member** record occurrences of **set-2** connected to the record assigned to user-defined indicator two.

Any member of **set-3** that is not in the intersection is disconnected from **set-3** during the execution of AOM.

If the member order of **set-3** is not sorted, then the set member order of **set-3** after AOM is a function of the order of record groups one and two.

It is not reasonable to use AOM unless record groups one and two have at least one record type in common. Otherwise, the intersection will be empty and **set-3** will have no members. **If set-3 is not $SYSSET, then it must allow each possible member record type for record groups one and two.**

The command status becomes 255 after successfully running AOM.

Examples of Command Usage

```
block/direct   .. E0 = AOM ("set-1,set-2,set-3")
block/indirect .. E0 = DMS ("AOM,set-1,set-2,set-3")
record/direct  .. E0 = AOM ("set-1,set-2,set-3")
record/indirect.. E0 = DMS ("AOM,set-1,set-2,set-3")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Boolean Commands

AOO — And of Owners with Owners

AOO

Command Arguments **AOO,set−1,set−2,set−3**

Currency Indicators

Used	Changed	Change Description
CU(1)	CM(set-3)	set to null
CU(2)	CRU	set to null

Description

Perform a Boolean intersection on record groups one and two and place the records common to both record groups in **set-3** as members.

Record group one consists of the **owner** record occurrences of **set-1** connected to the record assigned to user-defined indicator one.

Record group two consists of the **owner** record occurrences of **set-2** connected to the record assigned to user-defined indicator two.

Any member of **set-3** that is not in the intersection is disconnected from **set-3** during the execution of AOO.

If the member order of **set-3** is not sorted, then the set member order of **set-3** after AOO is a function of the order of record groups one and two.

It is not reasonable to use AOO unless record groups one and two have at least one record type in common. Otherwise, the intersection will be empty and **set-3** will have no members. **If set-3 is not $SYSSET, then it must allow each possible member record type for record groups one and two.**

The command status becomes 255 after successfully running AOO.

Examples of Command Usage

```
block/direct   .. E0 = AOO ("set-1,set-2,set-3")
block/indirect .. E0 = DMS ("AOO,set-1,set-2,set-3")
record/direct  .. E0 = AOO ("set-1,set-2,set-3")
record/indirect.. E0 = DMS ("AOO,set-1,set-2,set-3")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Boolean Commands

XMM — eXclude Members from Members — XMM

Command Arguments **XMM,set−1,set−2,set−3**

Currency Indicators

Used	Changed	Change Description
CU(1)	CM(set-3)	set to null
CU(2)	CRU	set to null

Description

Find the logical difference between record groups one and two and place the records exclusive to record group one in **set-3** as members.

Record group one consists of the **member** record occurrences of **set-1** connected to the record assigned to user-defined indicator one.

Record group two consists of the **member** record occurrences of **set-2** connected to the record assigned to user-defined indicator two.

Any member of **set-3** that is not in the logical difference is disconnected from **set-3** during the execution of XMM.

If the member order of **set-3** is not sorted, then the set member order of **set-3** after XMM is a function of the order of record groups one and two.

If set-3 is not $SYSSET, then it must allow each possible member record type for record groups one and two.

The command status becomes 255 after successfully running XMM.

Examples of Command Usage

```
block/direct    .. E0 = XMM ("set-1,set-2,set-3")
block/indirect .. E0 = DMS ("XMM,set-1,set-2,set-3")
record/direct   .. E0 = XMM ("set-1,set-2,set-3")
record/indirect.. E0 = DMS ("XMM,set-1,set-2,set-3")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Boolean Commands

XMO — eXclude Members from Owners — XMO

Command Arguments

XMO,set−1,set−2,set−3

Currency Indicators

Used	Changed	Change Description
CU(1)	CM(set-3)	set to null
CU(2)	CRU	set to null

Description

Find the logical difference between record groups one and two and place the records exclusive to record group one in **set-3** as members.

Record group one consists of the **member** record occurrences of **set-1** connected to the record assigned to user-defined indicator one.

Record group two consists of the **owner** record occurrences of **set-2** connected to the record assigned to user-defined indicator two.

Any member of **set-3** that is not in the logical difference is disconnected from **set-3** during the execution of XMO.

If the member order of **set-3** is not sorted, then the set member order of **set-3** after XMO is a function of the order of record groups one and two.

If set-3 is not $SYSSET, then it must allow each possible member record type for record groups one and two.

The command status becomes 255 after successfully running XMO.

Examples of Command Usage

```
block/direct   .. E0 = XMO ("set-1,set-2,set-3")
block/indirect .. E0 = DMS ("XMO,set-1,set-2,set-3")
record/direct  .. E0 = XMO ("set-1,set-2,set-3")
record/indirect.. E0 = DMS ("XMO,set-1,set-2,set-3")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Boolean Commands

XOM — eXclude Owners from Members — XOM

Command Arguments **XOM,set−1,set−2,set−3**

Currency Indicators

Used	Changed	Change Description
CU(1)	CM(set-3)	set to null
CU(2)	CRU	set to null

Description

Find the logical difference between record groups one and two and place the records exclusive to record group one in **set-3** as members.

Record group one consists of the **owner** record occurrences of **set-1** connected to the record assigned to user-defined indicator one.

Record group two consists of the **member** record occurrences of **set-2** connected to the record assigned to user-defined indicator two.

Any member of **set-3** that is not in the logical difference is disconnected from **set-3** during the execution of XOM.

If the member order of **set-3** is not sorted, then the set member order of **set-3** after XOM is a function of the order of record groups one and two.

If set-3 is not $SYSSET, then it must allow each possible member record type for record groups one and two.

The command status becomes 255 after successfully running XOM.

Examples of Command Usage

```
block/direct   .. E0 = XOM ("set-1,set-2,set-3")
block/indirect .. E0 = DMS ("XOM,set-1,set-2,set-3")
record/direct  .. E0 = XOM ("set-1,set-2,set-3")
record/indirect.. E0 = DMS ("XOM,set-1,set-2,set-3")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Boolean Commands

XOO — eXclude Owners from Owners — XOO

Command Arguments **XOO,set−1,set−2,set−3**

Currency Indicators

Used	Changed	Change Description
CU(1)	CM(set-3)	set to null
CU(2)	CRU	set to null

Description

Find the logical difference between record groups one and two and place the records exclusive to record group one in **set-3** as members.

Record group one consists of the **owner** record occurrences of **set-1** connected to the record assigned to user-defined indicator one.

Record group two consists of the **owner** record occurrences of **set-2** connected to the record assigned to user-defined indicator two.

Any member of **set-3** that is not in the logical difference is disconnected from **set-3** during the execution of XOO.

If the member order of **set-3** is not sorted, then the set member order of **set-3** after XOO is a function of the order of record groups one and two.

If set-3 is not $SYSSET, then it must allow each possible member record type for record groups one and two.

The command status becomes 255 after successfully running XOO.

Examples of Command Usage

```
block/direct    .. E0 = XOO ("set-1,set-2,set-3")
block/indirect .. E0 = DMS ("XOO,set-1,set-2,set-3")
record/direct   .. E0 = XOO ("set-1,set-2,set-3")
record/indirect.. E0 = DMS ("XOO,set-1,set-2,set-3")
```


© Copyright 1987 Micro Data Base Systems, Inc.

XIV. SPECIAL COMMANDS

The MDBS DML supports several commands that exist only with certain host languages or environments. These are called special commands.

Some special commands modify the data base control system. These commands perform functions such as:

- allocate the data base control system workspace for page buffers
- alter the end-of-set command status number from its default value of 255
- set user priorities in a multi-user environment.

Host programming languages that do not support program record structures also use special DML commands. These languages need a mechanism to transfer information between the application program and the data base control system. Three DML special commands exist to define and release data blocks for these host languages, enabling data transfer between the program and MDBS.DMS.

Another group of commands address those host programming languages do not allow direct DML command functions. These languages are indirect. They require a special variable to hold the command status of a DML command and another variable to hold the DMS command string. The DML supports two special commands to specify these command related variables.

The special DML commands that exist in your environment are listed in chapter seven of the MDBS III system specific installation manual.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Special Commands

COMMAND DETAILS

ALTEOS

ALTer End Of Set

ALTEOS

Command Arguments

ALTEOS (no arguments)

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The command status value of 255 that is normally used to indicate an end-of-set is altered within the scope of the application program that uses ALTEOS. If ALTEOS is permitted within a given host language, the system specific manual for that language indicates the altered end-of-set command status value (usually -1).

Examples of Command Usage

```
block/direct   .. E0 = ALTEOS (     )
block/indirect .. E0 = DMS ("ALTEOS")
record/direct  .. E0 = ALTEOS (     )
record/indirect.. E0 = DMS ("ALTEOS")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Special Commands

DBINIT **Data Base control system INITialization** **DBINIT**

Command Arguments **DBINIT, host language variables**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

When DBINIT is available, it is invoked as the first DML command of a run unit. Two or more host language variables or constants are used as arguments. The nature of these arguments and the precise actions performed by DBINIT are operating system specific. See the appropriate system specific manual. In general, DBINIT initializes the data base control system(s) to be used by a run unit.

In run units that support DBINIT, a single run unit can use DBINIT to have multiple MDBS data bases open simultaneously.

Examples of Command Usage

See system specific manuals.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Special Commands

DBLDR — Data Base control system LoaDeR — DBLDR

Command Arguments — **DBLDR, host language variables**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

Where available, DBLDR dynamically loads and unloads a copy of the data base control system into the central processor main memory (RAM) for a run unit. Three host language variables are used as arguments. These arguments are operating system and host language specific.

In run units that support DBLDR, a single run unit can use DBLDR repeatedly to open multiple MDBS data bases concurrently.

Examples of Command Usage

See system specific manuals.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Special Commands

DBSEL — Data Base SELection — DBSEL

Command Arguments — **DBSEL, host language variable**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

Where DBSEL is available, it can be used by a run unit to select any one of the concurrently open MDBS data bases for processing. The host language variable (or constant) is used to indicate which of the data bases is to be processed.

Examples of Command Usage

See system specific manuals.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Special Commands

DEFINE

DEFINE data block

DEFINE

Command Arguments

DEFINE,blk,list

Currency Indicators

Used	Changed	Change Description
none	none	

Description

This command is used with host languages that do not support program record types. It defines the **blk** data block to have the characteristics indicated in **list**. The composition of **list** is host language dependent. It typically consists of at least a list of the host language variables that make up the data block. Full details for each host language appear in the system specific manuals.

A data block must have been defined with the DEFINE command before it can be used as an argument in any other DML command. A given host language variable can appear more than once in the same or different data blocks. Many different data blocks can be used in a host language program. Only the last data block defined before using DBOPN will exist after invoking DBOPN. Any data block defined after the data base is opened will continue to exist until it is undefined (with the UNDEF command).

Examples of Command Usage

```
block/direct   .. E0 = DEFINE ("blk",list)
block/indirect .. E0 = DMSD ("DEFINE,blk",list)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Special Commands

DMSSJP — DMS Set Jump — DMSSJP

Command Arguments — **DMSSJP, host language jump description, command status array**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

This special command pertains to C host languages and provides a concise error trapping mechanism. The indicated host language jump description is a pointer to a stack environment. The values of the indicated array specify which command status numbers are to be exempted from the error trapping. The last element of this array must have a value of 0.

When DMSSJP is invoked, its first argument indicates the environment to which control is transferred whenever a command status other than those specified in the array occurs. This condition remains in force until DMSSJP is re-invoked with a different argument (indicating a different error handling module or a different set of exempt command status numbers). If the first argument is 0, then standard error handling is resumed. If the second argument is 0 rather than an array, then all command status numbers other than 0 and 255 are trapped.

Examples of Command Usage

See system specific manuals.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Special Commands

EXTEND EXTEND data block EXTEND

Command Arguments EXTEND,blk,list

Currency Indicators

Used	Changed	Change Description
none	none	

Description

This command is used with host languages that do not support program record types. It is used to extend the previously defined **blk** data block by assigning more host language variables to it. These additional variables are indicated in **list** and they will follow the variables already defined for the data block. The composition of **list** is host language dependent. It typically consists of at least a list of the additional variables. Full details for each host language appear in the system specific manuals.

The **blk** data block must have been defined with the DEFINE command before it can be used in an EXTEND command otherwise a command status of 7 is returned. A given host language variable can appear more than once in the same or different data blocks.

Examples of Command Usage

```
block/direct    .. E0 = EXTEND ("blk",list)
block/indirect .. E0 = DMSD ("EXTEND,blk",list)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Special Commands

MPL — Multiuser Priority Level — MPL

Command Arguments **MPL, iblk**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The MPL command is of interest in multiuser processing situations. It enables different run units to have varying degrees of throughput priority. If a run unit invokes MPL, the value of **iblk**'s host language variable is used to set that run unit's priority. The **iblk** host language variable must be consistent with a two byte unsigned data item. Its value can be any integer from 1 through 10.

The highest possible priority level is 1. Selecting this level results in throughput that is at least as fast as level 2. Level 2 throughput, in turn, is at least as fast as level 3, and so forth. A run unit that does not require fastest throughput should use MPL to set its priority level lower than 1 (i.e., 2-10). A run unit that requires more than minimal throughput should use MPL to set its priority level higher than 10 (i.e., 9-1). A run unit can invoke MPL repeatedly to alter its own priority level at desired points in the program.

An interactive run unit has a priority level of 3 until it invokes MPL to get a different priority. A batch run unit has a priority level of 7 until it invokes MPL to get a different priority.

Examples of Command Usage

```
block/direct   .. E0 = MPL ("iblk")
block/indirect .. E0 = DMS ("MPL,iblk")
record/direct  .. E0 = MPL (iblk)
record/indirect.. E0 = DMSD ("MPL",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Special Commands

SETPBF **SET Program BuFfer region** **SETPBF**

Command Arguments **SETPBF,list**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

This command allocates the region of main memory that is to be used for program buffers. If SETPBF is invoked, it must appear before any other DML command that needs to use program buffers. The composition of **list** is environment dependent and is fully discussed in the various system specific manuals. It typically consists of a starting address and a length (in bytes) for the page buffer area. For instance, with FORTRAN the starting address could be indicated by an array name and the length would be the number of bytes occupied by that array. In the C language, for example, the SBRK function is a dynamic memory allocation function; the value returned by SBRK can be used as an input to the SETPBF DML command.

In many environments the programmer can optionally omit SETPBF. If omitted in a single-user environment, MDBS.DMS itself sets up the largest program (i.e., page) buffer region possible. If omitted in a multiuser environment, MDBS.DMS allocates adequate program buffers. For a given operating environment, the system specific manual indicates whether SETPBF can be omitted.

Examples of Command Usage

See system specific manuals.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Special Commands

UNDEF — UNDEFine data blocks — UNDEF

Command Arguments — **UNDEF (no arguments)**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

All data blocks previously defined in the program are undefined (i.e., they no longer exist). This is typically employed in host language programs that use chaining. UNDEF is invoked before calling a chained subprogram; the needed data blocks are redefined within the subprogram.

Examples of Command Usage

```
block/direct   .. E0 = UNDEF (     )
block/indirect .. E0 = DMS ("UNDEF")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Special Commands

VARCMD **Variable for Command string** VARCMD

Command Arguments **VARCMD, host language variable**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

This special command is used in some host languages to specify a host language variable whose value is treated as a DML command string. The application program sets the value of this variable to the desired command string (i.e., DML command and arguments) prior to each invocation of the data base control system.

Examples of Command Usage

See system specific manuals.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Special Commands

VARCS — VARiable for Command Status — VARCS

Command Arguments — **VARCS, host language variable**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The indicated host language variable will hold the command status that results from invoking a DML command. VARCS is needed only in those very few host languages that do not permit function subprograms. If VARCS is available for a host language (see system specific manuals), it should be the first DML command used in an application program. An exception to this is the case where DBINIT is also available, in which case VARCS is invoked after DBINIT.

Examples of Command Usage

See system specific manuals.

© Copyright 1987 Micro Data Base Systems, Inc.

XV. MULTIUSER LOCKING COMMANDS

In a multiuser environment, conflicts can occur when two or more applications attempt to use, modify, or delete a record occurrence in the data base. To co-ordinate the information usage from the data base and reduce these conflicts, MDBS III supports two lock mechanisms:

- Passive Locks

  A passively locked record is any record occurrence currently in use by an application. That is, the record is assigned to one of the following currency indicators:

  - the current of run unit
  - the current owner of a set
  - the current member of a set
  - the current record for any user-defined indicator

  A passively locked record is available for data retrieval by any run unit, however, it is modifiable only by the run unit applying the passive lock. Two or more run units can passively lock a single record occurrence concurrently.

- Active Locks

  An actively locked record is any record currently in use by an application that has a DML locking command in effect for the record type or the currency indicator referencing the record. The data base control system has three active lock mechanisms that differ in scope and affect the following items:

  - A single record (the current of run unit or a user-defined indicator)
  - All records of a single record type
  - A set (its current owner and current member)

  An actively locked record is accessible only to the application that issues the active lock; another run unit cannot retrieve or modify that record while the lock is on and the record is current.

The effects of passive and active locks on the DML command groups (record retrieval, modification, creation, connection, disconnection and deletion) are presented in Table XV-1.

In a multiuser environment, a record can be read if it is not locked or if it is passively locked. A record cannot be read if it is actively locked. A run unit can modify a record if that record is not locked by any other run unit. As long as a record is actively or passively locked by one run unit, no other run unit can modify it.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Multiuser Commands

The application programmer for a multiuser environment should keep the following facts in mind:

- Use currency indicators sparingly. Null out currency indicators you are not presently using since they passively lock the records they reference

- All records that are not actively locked can be read (retrieved) by any run unit

- Remove active locks as quickly as possible after completing the data base transactions. While a record is actively locked, the data base control system prevents other run units from these activities:

  - finding the record
  - retrieving the record
  - modifying the record
  - navigating through the record (if a program encounters an actively locked record while trying to find another record, the locked record blocks the find command from proceeding through the set.)

- A run unit can modify a record occurrence with either of the following situations:

  - the run unit has an active lock on the record occurrence
  - only that run unit has a passive lock on the record occurrence

- A record occurrence that is passively locked by more than one run unit cannot be modified or deleted by any run units

- MDBS QRS uses only passive locks

The data base control system provides automatic retry capability for instances when a run unit attempts to access a locked record occurrence (or set) and the access is initially denied. MDBS.DMS re-performs the attempted command until the access succeeds or a limit of consecutive access failures is reached. In the latter case, MDBS.DMS returns a command status message to the program indicating an encounter with an active lock.

The DML multiuser contention count command (MCC) specifies the retry time interval length and the retry limit.

As soon as a run unit closes the data base, the data base control system clears all currency indicators and all locks set by the application.

Conflicts caused by the multiuser environment do not affect user security restrictions (read and write access for records, data items, and sets specified in the DDL source specification). The data base control system automatically enforces all security restrictions independently of the environment (single user or multiuser).

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Multiuser Commands

The multiuser data base control system does not support page image posting.

Multiuser locking commands used in single user environments have no effect.

A special DML command for setting run unit priority levels is available in some environments (see Chapter XIII).

Table XV-1. Multiuser Locking Privileges

Run Unit A: uses lock type	Run Unit B: Find or retrieve the locked record	Modify the locked record	Create a record	Connect a record into set-2	Disconnect a record from set-2	Disconnect all member(owner) records from set-2	Delete the locked record
Passive lock on a record	Yes	No Note 1	N/A	Yes	Yes Note 2	Yes Note 3	No
Active lock on a record	No	N/A	N/A	N/A	N/A	Yes	N/A
Active lock on the type of record	No	N/A	Yes	N/A	N/A	Yes	No
Active lock on set-1	Yes Note 4	Yes Note 5	Yes Note 6	Yes Note 7	Yes Note 7	Yes Note 7	No Note 8

Notes:

1. **General**
   A record occurrence that is passively locked by two or more run units cannot be modified.

   Exception
   The data base environment command (DBENV) with its fourth option shuts off passive locking checks and allows a run unit to modify the record. See the DBENV command in chapter twelve for more information.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Multiuser Commands

2. **General**

A record occurrence that is passively locked by two or more run units can be disconnected from a set (set-2) by a run unit.

Exception

A record that is passively locked by two or more run units cannot be disconnected from that set if the following conditions exist:

- Two run units share the same current set owner record
- The same two run units share the same current set member record

3. **General**

A run unit can disconnect all set members for a specific set owner record or disconnect all set owners for a specific set member record, even though one of the record occurrences to be disconnected is passively locked by another run unit.

Exception

The set disconnect commands (remove the set members from a particular set owner record or remove the set owners from a particular set member record) will not function if the following conditions exist:

- Two run units share the same current record. In removing set members, the run units share the same current owner of the set. In removing set owners, the run units share the same current member of the set.

- The run unit NOT performing the disconnect command has a record occurrence in the other currency indicator for the set . In removing set members, the member of the set involved is not null; in removing set owners, the owner of the set is not null.

4. **General**

For a record that is locked with an active set lock, the record occurrence can be accessed by run units that did not issue the lock by using another set.

Exception

The record occurrence is inaccessible to all run units via the locked set except the run unit that issued the lock.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Multiuser Commands

5. **General**
Any run unit can modify one or more data items in a record occurrence locked by an active set lock.

Exception
A data item value that is part of the sort key of the locked set can be modified only by the run unit that issued the active set lock.

6. **General**
A new record can be created by a run unit while another run unit has a set actively locked.

Exception
A new record cannot be created if the record is automatically inserted (as a set owner or member) into a set that is locked by another run unit.

7. **General**
Any run unit can connect a record occurrence to a set or disconnect a record occurrence from a set when the record occurrence is locked by another run unit with an active set lock.

Exception
A record occurrence cannot be connected to a set or disconnected from a set if the set is locked by another run unit.

8. **General**
A run unit can delete a record occurrence from the data base while another has a set actively locked.

Exception
A record occurrence cannot be deleted from a data base if it is the current member or current owner of a set that is actively locked by another run unit.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Multiuser Commands

COMMAND DETAILS

MAU **Multiuser Active User indicators** **MAU**

Command Arguments **MAU,iblk**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

One host language variable is defined for **iblk**. It must be consistent with a one byte unsigned data item. The value of that variable must be an integer between 0 and 255 inclusive. This value indicates how many user-defined indicators are to be made actively locked, rather than passively locked. If the value is 0, then all user-defined indicators revert to their normal state of being passively locked. If the value is 12, then user indicators 1 through 12 are actively locked and all other allocated user indicators are passively locked. MAU can be used repeatedly within a run unit to alter the number of user indicators that are actively locked.

A user indicator must have been allocated (with AUI) before it can be declared to be actively locked. A command status message of 62 or 63 results from MAU if any record that is the current record of a user indicator is already actively or passively locked by another run unit. In this case, MAU will not affect the current active/passive status of user indicators.

Examples of Command Usage

```
block/direct   .. E0 = MAU ("iblk")
block/indirect .. E0 = ("MAU,iblk")
record/direct  .. E0 = MAU (iblk)
record/indirect.. E0 = ("MAU",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Multiuser Commands

MCC **Multiuser Contention Count** MCC

Command Arguments **MCC,iblk**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

This command sets the number of re-tries and the time units between tries that will be enforced by MDBS.DMS when an access attempt is prohibited due to multiuser locking. This is enforced for all DML commands that follow MCC until another MCC is invoked. The first host language variable for **iblk** has an integer value that indicates the number of re-tries. The second variable defined for **iblk** has an integer value that indicates the number of time units between tries. The time unit is operating system dependent (see system specific manuals), but is never smaller than .01 seconds. Both of **iblk**'s variables must be consistent with an unsigned data item that is two bytes in size.

Examples of Command Usage

```
block/direct   .. E0 = MCC ("iblk")
block/indirect .. E0 = DMS ("MCC,iblk")
record/direct  .. E0 = MCC (iblk)
record/indirect.. E0 = DMSD ("MCC",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Multiuser Commands

MCF **Multiuser Current of run unit Free** MCF

Command Arguments **MCF (no arguments)**

Currency Indicators

Used	Changed	Change Description
CRU	none	

Description

If the record that is the current of run unit has been actively locked by MCP, invoking MCF will free that record so that it is only passively locked.

Examples of Command Usage

```
block/direct   .. E0 = MCF (     )
block/indirect .. E0 = DMS ("MCF")
record/direct  .. E0 = MCF (     )
record/indirect.. E0 = DMSD ("MCF")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Multiuser Commands

MCP — Multiuser Current of run unit Protect — MCP

Command Arguments

MCP (no arguments)

Currency Indicators

Used	Changed	Change Description
CRU	none	

Description

When this command is executed, the record that is the current of run unit is actively locked for as long as it is the current of run unit. After MCP has been invoked (and before invoking MCF), a record can become the current of run unit only if it is not actively or passively locked by some other run unit. When it does become the current of run unit, it becomes actively locked and the former current of run unit is no longer actively locked (by way of MCP).

To reinstate the current of run unit to its normal condition of being only passively locked, the MCF command is used.

Examples of Command Usage

```
block/direct   .. E0 = MCP (     )
block/indirect .. E0 = DMS ("MCP")
record/direct  .. E0 = MCP (     )
record/indirect.. E0 = DMSD ("MCP")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Multiuser Commands

MRID **Multiuser Run unit IDentification** **MRID**

Command Arguments **MRID,oblk**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

MRID identifies run units that cause multiuser locking conflicts from active and passive locks.

The **oblk** is consistent with an eight byte binary item and contains four pieces of information that identify the run unit requesting the information and the run unit that is causing the lock. Table XV-2 describes the structure of the variable within **oblk**.

Table XV-2. Data Variable Structure for MRID Command

Byte Numbers	Description
1	Internal ID of requesting run unit
2 - 4	External ID of requesting run unit
5	Internal ID of blocking run unit
6 - 8	External ID of blocking run unit

Examples of Command Usage

```
block/direct   .. E0 = MRID ("oblk")
block/indirect .. E0 = DMS ("MRID,oblk")
record/direct  .. E0 = MRID ("oblk")
record/indirect.. E0 = DMS ("MRID,oblk")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Multiuser Commands

MRTF **Multiuser Record Type Free** **MRTF**

Command Arguments **MRTF,rec (rec is optional)**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

All record occurrences of **rec** are freed. They are no longer actively locked by the run unit. The exception is that if one of those occurrences is actively locked through some other multiuser protect command, then it is still actively locked (until the corresponding multiuser free command is invoked). If no record type is specified with this command, then every record type that is actively locked by the run unit is freed.

Examples of Command Usage

```
block/direct   .. E0 = MRTF ("rec")
block/indirect .. E0 = DMS ("MRTF,rec")
record/direct  .. E0 = MRTF ("rec")
record/indirect.. E0 = DMS ("MRTF,rec")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Multiuser Commands

MRTP **Multiuser Record Type Protect** **MRTP**

Command Arguments **MRTP,rec**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

All occurrences of the record type **rec** are actively locked by the run unit, if no other run unit has locked any of those occurrences (either actively or passively). If any occurrence of **rec** is locked by another run unit, then the MRTP request is refused and a command status message to that effect is returned.

Examples of Command Usage

```
block/direct   .. E0 = MRTP ("rec")
block/indirect .. E0 = DMS ("MRTP,rec")
record/direct  .. E0 = MRTP ("rec")
record/indirect.. E0 = DMS ("MRTP,rec")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Multiuser Commands

MSF — Multiuser Set Free — MSF

Command Arguments MSF,set-1 (set-1 is optional)

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The indicated **set-1** is freed if it has been actively locked earlier in the run unit. The record occurrences of the owner and member record types are no longer actively locked, unless there is also an active lock on one or more of those record types or on a record occurrence (until the corresponding multiuser free command is invoked). If no set is specified, all actively locked sets of the run unit are freed.

Examples of Command Usage

```
block/direct   .. E0 = MSF ("set-1")
block/indirect .. E0 = DMS ("MSF,set-1")
record/direct  .. E0 = MSF ("set-1")
record/indirect.. E0 = DMS ("MSF,set-1")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Multiuser Commands

MSP — Multiuser Set Protect — MSP

Command Arguments **MSP,set-1**

Currency Indicators

Used	Changed	Change Description
none	none	

Description

An active lock is placed on **set-1** if no other run unit has a lock (either active or passive) on both an owner record and a member record of that set. If any such occurrence is locked by another run unit, then the MSP request is refused and a command status message to that effect is returned.

Examples of Command Usage

```
block/direct   .. E0 = MSP ("set-1")
block/indirect .. E0 = DMS ("MSP,set-1")
record/direct  .. E0 = MSP ("set-1")
record/indirect.. E0 = DMS ("MSP,set-1")
```


© Copyright 1987 Micro Data Base Systems, Inc.

XVI. RECOVERY COMMANDS

The features and commands described in this chapter are available in various versions of the Recovery and Transaction Logging (RTL) form of the MDBS data base management system. The RTL form of the data base control system provides the same DML command conventions as the standard MDBS.DMS. Since MDBS.RTL performs additional functions, it requires a larger minimum buffer size than the standard data base control system.

The transaction logging and recovery DML commands focus on two capabilities that can be used individually or in combination to provide added protection against hazards to the data base:

- Page image file posting
  - Declare a page image file -- PIFD
  - Abort a complex transaction in a page image file -- TRABT
  - Begin a complex transaction -- TRBGN
  - Commit or Close a complex transaction -- TRCOM

- Transaction logging
  - Declare a transaction logging file -- LGFILE
  - Flush the log file buffer -- LGFLSH
  - Write a message to the log file -- LGMSG
  - Begin a complex transaction -- TRBGN
  - Commit or Close a complex transaction -- TRCOM

Page Image File Posting

Posting data base transactions with a page image file reduces the vulnerability of a data base to external factors such as power failures. The page image file holds data base changes, either a single DML command or a series of DML commands that comprise a complex transaction. When the change or complex transaction completes, MDBS.RTL flushes the page image file and incorporates the changes into the data base.

The page image file reduces the amount of time the data base is inconsistent. That is, posting transactions to the page image file minimizes the amount of time the data base page buffers in RAM differ with the data base on disk. So, if a power failure occurs midway through a complex transaction and the page image file is used, the data base will remain consistent. This helps prevent data base inconsistency.

Page image posting allows the application developer to specify the beginning of a complex transaction sequence (a sequence of logically related data base changes). A transaction sequence can be aborted or committed to the data base. Aborting a complex transaction prevents any portion of the complex transaction from being applied to the data base. The data base remains consistent; it is current up to the start of the complex transaction.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Recovery Commands

The page image file must be declared before opening the data base to allow MDBS.RTL to post transactions to the page image file. Use a page image file of sufficient size; otherwise, the data base could become inconsistent if a transaction sequence is abnormally interrupted. (This event produces a command status 15, data base not closed previously, the next time the data base is opened.)

Note: Page image posting is only available for single user environments.

Transaction Logging

Transaction logging provides the primary recovery capability from inconsistent states of the data base and from external factors (power failures, hardware malfunctions, and erroneous data modification) by applying recorded data base changes (stored as commands with associated data) contained in a log file to a back up copy of the data base. Specifically, MDBS.RTL records all data base transactions to a separate log file. The recorded transactions begin at a specific point in time and correspond to a copy of the data base at the same point in time. The Recovery utility (provided with the RTL form) applies transactions from the log file to the back-up copy of the data base, recreating an up-to-date data base.

The transaction logging form of the data base control system provides DML commands to specify and maintain a log file, write messages to the log file (such as a time stamp), and to define complex transactions.

The MDBS Recovery utility (RCV) processes log file transactions with a back up copy of a data base, resulting in a consistent, up to date data base. RCV allows selected transactions or the entire log file contents to be applied to the data base.

The log file recognizes and stores complex transactions recorded by MDBS.RTL. In processing a complex transaction, RCV verifies from the log file that the complex transaction is committed (complete). RCV processes committed transactions. RCV does not apply uncommitted (incomplete) complex transaction; it ignores any logged transactions that occur after an TRBGN, if a subsequent TRCOM was not reached during a run unit.

A complete description of the RTL form is presented in the MDBS- RTL Manual.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Recovery Commands

COMMAND DETAILS

LGFILE — LoG FILE specification

LGFILE

Command and Arguments LGFILE,iblk

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The fully qualified name of a file to which transactions can be logged is specified by iblk's host language variable. This variable must be consistent with a string data item, whose maximum length is operating system dependent (see the appropriate system specific manual for size information). If the file to which transactions are to be logged is the same as the log file specified in the DDL specification, this command need not be invoked. If it is invoked, this command must appear before the DBOPN command.

Examples of Command Usage

```
block/direct   .. E0 = LGFILE ("iblk")
block/indirect .. E0 = DMS ("LGFILE,iblk")
record/direct  .. E0 = LGFILE (iblk)
record/indirect.. E0 = DMSD ("LGFILE",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Recovery Commands

LGFLSH — LoG file buffer FLuSH — LGFLSH

Command and Arguments LGFLSH (no arguments)

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The buffers holding transactions to be logged are flushed to the log file when this command is invoked. LGFLSH is automatically invoked by DBSAVE. This is especially useful when an important transaction has been processed by the system. It can be written to the log file before the log file buffer is full.

Examples of Command Usage

```
block/direct   .. E0 = LGFLSH (     )
block/indirect .. E0 = DMS ("LGFLSH")
record/direct  .. E0 = LGFLSH (     )
record/indirect.. E0 = DMS ("LGFLSH")
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Recovery Commands

LGMSG — LoG file MeSsaGe

LGMSG

Command and Arguments LGMSG,iblk

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The message specified in iblk's host language variable is written to the log file. This variable must be consistent with a string data item not exceeding 90 bytes. Messages written to the log file can be listed using the RCV utility program. The use of the LGMSG command is especially useful for surveillance.

Examples of Command Usage

```
block/direct   .. E0 = LGMSG ("iblk")
block/indirect .. E0 = DMS ("LGMSG,iblk")
record/direct  .. E0 = LGMSG (iblk)
record/indirect.. E0 = DMSD ("LGMSG",iblk)
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Recovery Commands

PIFD — Page Image File Declaration — PIFD

Command and Arguments PIFD,iblk

Currency Indicators

Used	Changed	Change Description
none	none	

Description

The fully qualified file name, indicated by iblk's host language variable, is declared to be a page image file for use by MDBS.DMS in page image posting. This command must be invoked prior to use of the TRBGN, TRCOM, and TRABT commands. The host language variable must be consistent with a string data item. The size (in bytes) is operating system dependent and is documented in the corresponding system specific manual. If PIFD is invoked, it must appear before the data base is opened.

Examples of Command Usage

```
block/direct   .. E0 = PIFD ("iblk")
block/indirect .. E0 = DMS ("PIFD,iblk")
record/direct  .. E0 = PIFD (iblk)
record/indirect.. E0 = DMSD ("PIFD",iblk)
```


© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Recovery Commands

TRABT — TRansaction ABorT — TRABT

Command and Arguments TRABT (no arguments)

Currency Indicators

Used	Changed	Change Description
none	most currency indicators are set to null	

Description

Posting Effects (occur only if PIFD has been invoked):
This command aborts a complex transaction sequence that was initiated by the TRBGN command. A transaction cannot be aborted after it has already been committed (with the TRCOM command). This command invokes NCI internally to ensure that no inconsistent currency indicators will be present.

Logging Effects:
The transaction sequence that is aborted will be ignored by the RCV recovery processor, but TRABT cannot be invoked if the application program does not use the page-image file.

Examples of Command Usage

```
block/direct   .. E0 = TRABT (    )
block/indirect .. E0 = DMS ("TRABT")
record/direct  .. E0 = TRABT (    )
record/indirect.. E0 = DMS ("TRABT")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Recovery Commands

TRBGN **TRansaction BeGiN** TRBGN

Command and Arguments TRBGN (no arguments)

Currency Indicators

Used	Changed	Change Description
none	none	

Description

Posting Effects (occur only if PIFD has been invoked):
This command denotes the beginning of a complex transaction , a sequence of several DML commands that logically comprise a transaction to an end user.

Corresponding to the complex transaction beginning are the complex transaction abort and the complex transaction commit. The complex transaction abort (TRABT) signals the end of the transaction and that the DML commands within the complex transaction are not to become part the data base. The complex transaction commit (TRCOM) singals the end of the complex transaction and permanently applies the data base changes of the complex transaction to the data base.

After TRCOM or TRABT has been invoked, TRBGN must be re-invoked to initiate another complex transaction. If TRBGN is re-invoked without having issued a TRABT or TRCOM command, the data base control system returns a nonzero command status and ignores the re-invocation of TRBGN.

Logging Effects:
This command denotes the beginning of a complex transaction , a sequence of several DML commands that logically comprise a transaction to an end user.

The data base control system records all individual DML commands that change the data base and all complex transaction commands (begin, abort, and commit). During the data base recovery, however, RCV ignores any transactions that follow TRBGN unless a subsequent TRCOM command exists in the log file.

The TRBGN and TRCOM commands are used in tandem to ensure the data base recovery either performs the entire complex transaction (with all of the DML commands comprising the complex transaction) or ignores the complex transaction (if the TRCOM is not present).

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Recovery Commands

Examples of Command Usage

```
block/direct   .. E0 = TRBGN (     )
block/indirect .. E0 = DMS ("TRBGN")
record/direct  .. E0 = TRBGN (     )
record/indirect.. E0 = DMS ("TRBGN")
```

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Recovery Commands

TRCOM

TRansaction COMmit

TRCOM

Command and Arguments TRCOM (no arguments)

Currency Indicators

Used	Changed	Change Description
none	none	

Description

Posting Effects (occur only if PIFD has been invoked):
The complex transaction commit (TRCOM) signals the end of the complex transaction (a sequence of DML commands that modify the data base) and permanently applies the data base changes of the complex transaction to the data base. It has the added effect of clearing the contents of the page image file declared with the PIFD command.

Logging Effects:
This command indicates the end of a complex transaction. Any transactions logged since the last TRBGN command was invoked can now be processed by the recovery utility (RCV) while restoring the data base.

Examples of Command Usage

```
block/direct    .. E0 = TRCOM (     )
block/indirect .. E0 = DMS ("TRCOM")
record/direct   .. E0 = TRCOM (     )
record/indirect.. E0 = DMS ("TRCOM")
```


© Copyright 1987 Micro Data Base Systems, Inc.

XVII. COMMAND STATUS DESCRIPTIONS

This chapter describes the different command status messages that may result from the execution of a DML command. For each command status (other than 0 and 255), the possible reasons for which that command status could be obtained are described. In almost all cases, the reasons themselves suggest a solution to the problem.

COMMAND STATUS DETAILS

**** -1**** End of Set **** -1****

This command status is identical to command status 255. See the command status 255 entry for descriptive information.

****1**** Invalid area name ****1****

Possible Causes:

1. Typographical error in spelling area name.
2. Incorrect order in command string.
3. Command string not terminated properly.

****2**** Invalid set name ****2****

Possible Causes:

1. Typographical error in spelling set name.
2. Incorrect order in command string.
3. Command string not terminated properly.

****3**** Invalid record type name ****3****

Possible Causes:

1. Typographical error in spelling record type name.
2. Incorrect order in command string.
3. Command string not terminated properly.

****4**** Invalid data item for this record type ****4****

Possible Causes:

1. Typographical error in spelling data item name.
2. Incorrect order in command string.
3. Command string not terminated properly.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Statuses

5 Invalid owner record type for this set **5**

Possible Causes:

1. The record occurrence specified has not been defined as the owner for the set specified.
2. Typographical error in spelling set name.

6 Invalid member record type for this set **6**

Possible Causes:

1. The record occurrence specified has not been defined as the member for the set specified.
2. Typographical error in spelling set name.

7 Invalid data block name **7**

Possible Causes:

1. A data block specified in the command string has not been defined.
2. A blank data block name was specified for a "DEFINE" command.
3. An "EXTEND" command does not immediately follow a "DEFINE" command.

8 Invalid name **8**

Possible Causes:

1. There was a syntax error related to the filename specification in a DBOPN or DBOPNA command.
2. Invalid drive specification.

9 Invalid number **9**

Possible Causes:

1. An invalid number was passed to DBCNV or DBENV.

10 Syntax error on command line **10**

Possible Causes:

1. Self-explanatory.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Statuses

****11**** Record already member/owner of set ****11****

Possible Causes:

1. An attempt was made to connect a member record to an owner record for a set, when that member was already connected to that owner for that set.
2. An attempt was made to connect an owner record to a member record for a set, when that owner was already connected to that member for that set.

****12**** Record does not belong to set occurrence ****12****

Possible Causes:

1. The record occurrence specified by the SME (or SOE) command is not a member or owner of the specified set.

****13**** Invalid area for this record type ****13****

Possible Causes:

1. An area has been specified in the CRA command and the record type does not belong to this area.

****14**** Data base area already open ****14****

Possible Causes:

1. The DBOPN or DBOPNA (for a given area) command was invoked twice.
2. For interpretive languages, DBCLS or DBCLSA was not called.

****15**** Data base not closed previously ****15****

Possible Causes:

1. The data base may be inconsistent since it was not closed previously.

****16**** Area not consistent with main area ****16****

Possible Causes:

1. A referenced area was not consistent with the main area.
2. An old copy of the area was on-line rather than a copy of the area that is up-to-date with the on-line main area.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Statuses

17 No space available in data base **17**

Possible Causes:

1. No space is available in the data base or area for processing the requested transaction.

18 Set internally inconsistent **18**

Possible Causes:

1. This command status is very rare and is returned from the following commands: CRA, CRS, IMS, IOS.
2. This command status occurs due to insufficient space being available for pointers to be allocated.

19 No duplicate record allowed **19**

Possible Causes:

1. No duplicates have been allowed in the DDL specifications for the record type.

20 No current of run unit **20**

Possible Causes:

1. Self-explanatory.
2. The user should check the currency indicators changed by the previous DML command.

21 No current owner of set type **21**

Possible Causes:

1. The command SOC, SOE, SOM, SOO or SOU was not invoked for this set.
2. The command FFO, FLO, FNO, FPO, SMC, SMM, SMO or SMU returned command status 255 previously.

22 No current member of set type **22**

Possible Causes:

1. The command SMC, SME, SMM, SMO or SMU was not invoked for this set.
2. The command FFM, FLM, FNM, FPM, SOC, SOM, SOO or SOU returned command status 255 previously.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Statuses

****23**** No current record of record type ****23****

Possible Causes:

1. The command CRS was not previously invoked for this record type.
2. The command DRC, DRM, or DRO was invoked for the current record of this record type.

****24**** No current user record ****24****

Possible Causes:

1. There is no current record of a referenced user-defined indicator.

****25**** User-defined currency indicator not allocated ****25****

Possible Causes:

1. A referenced user-defined indicator has not been allocated.

****26**** Current of run unit not in specified area ****26****

Possible Causes:

1. The command FNS has been invoked with an area specified as an argument, and the current of run unit is not in the indicated area.

****27**** Inappropriate set ****27****

Possible Causes:

1. An inappropriate set appeared as an argument. For example, the last set specified in a Boolean command is not system- owned.
2. The first or second set specified with a Boolean command is N:l.

****28**** Inappropriate record type ****28****

Possible Causes:

1. An FRK command has been invoked on a record whose record type has not been declared as having a calc key.

****29**** Cannot obtain based on record type ****29****

Possible Causes:

1. An attempt was made to obtain a record based on a sort key involving the record type names. Use the corresponding find command and GETC instead.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Statuses

30 Data out of range **30**

Possible Causes:

1. The data being input is outside the range of possible values as indicated in the DDL specifications.
2. The value of an input argument to a DML command is not a permissible value for that argument.

31 Insufficient room in memory **31**

Possible Causes:

1. For languages using the SETPBF command, not enough buffer space has been allocated.
2. For languages not using the SETPBF command, the DMS has been ORGed too high in memory, or the FIRST and LAST word in memory do not allow enough room.

32 Data base opened for read access only **32**

Possible Causes:

1. Self-explanatory.
2. Invalid read/write specifications in OPEN command.

33 Data conversion error **33**

Possible Causes:

1. An error has been obtained during data conversion by MDBS.DMS based on the DDL item specifications.

34 No such routine **34**

Possible Causes:

1. Typographical error in spelling command name.
2. Logging commands have been used without the RTL form of MDBS.
3. A command was invoked using wrong entry point (DMS or DMSD).
4. A command was invoked that attempted to perform explicitly disabled processing (e.g., disabled by NOCALC or NOFLOAT).

35 Duplicate data block name **35**

Possible Causes:

1. The DEFINE command was invoked for a data block that already exists.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Statuses

****36**** Invalid password ****36****

Possible Causes:

1. User name/password does not exactly match DDL specifications.

****37**** Invalid number of arguments ****37****

Possible Causes:

1. Too many arguments specified in command string.

****38**** Data base not open ****38****

Possible Causes:

1. A DML command, needing to access the data base, has been invoked before the data base was opened.

****39**** Set not sorted ****39****

Possible Causes:

1. The FOSK or FMSK command is invoked on a non-sorted set.

****40**** User may not read this set ****40****

Possible Causes:

1. The user's read access codes have no common elements with the read access codes of the set being processed.

****41**** User may not write this set ****41****

Possible Causes:

1. The user's write access codes have no elements in common with the write access codes of the set being processed.

****42**** User may not read this record ****42****

Possible Causes:

1. The user's read access codes have no elements in common with the read access codes of the record being processed.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Statuses

****43**** User may not write this record ****43****

Possible Causes:

1. The user's write access codes have no common elements with the write access codes of the record being processed.

****44**** User may not read this item ****44****

Possible Causes:

1. The user's read access codes have no elements in common with the read access codes of the item being processed.

****45**** User may not write this item ****45****

Possible Causes:

1. The user's write access codes have no elements in common with the write access codes of the item being processed.

****48**** No value labels defined for this field ****48****

Possible Causes:

1. The specified field presently has no value labels.

****49**** Invalid data base key ****49****

Possible Causes:

1. The value input to the SCD command is not the data base key for a valid record occurrence in the data base.
2. A record whose data base key was formerly saved by SDC may have since been deleted or moved.

****50**** Checksum error ****50****

Possible Causes:

1. An error was encountered when a checksum was being performed on the data base pages.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Statuses

****51**** Disk read or write error ****51****

Possible Causes:

This command status is returned as a result of any I/O error. Some of the possible causes are:

1. Bad sector on disk.
2. Disk drive misaligned.
3. The operating system has detected an error when trying to read/write from/to disk.

****52**** Invalid data base format ****52****

Possible Causes:

An error was encountered when the system performed a synchronization check on the data base pages. This may mean that:

1. Disk is in the wrong drive.
2. The correct area disk is not on-line.

****53**** Data base not expanded ****53****

Possible Causes:

1. The data base was initialized with the **s** or **o** option, but was not expanded with MDBS.DMU.

****54**** File not present ****54****

Possible Causes:

1. Disk in wrong drive.
2. File name misspelled.

****55**** Log file I/O error ****55****

Possible Causes:

1. An I/O error was encountered when trying to open the log file.
2. See also Command Status 51.

****56**** Log file not current ****56****

Possible Causes:

1. The log file is not up-to-date.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Statuses

****57**** Logging inhibited ****57****

Possible Causes:

1. An I/O error was detected on the log file.
2. The log file may already be open.

****59**** Needed area not on-line ****59****

Possible Causes:

1. As the result of a DML command, the data base control system attempted to reference an area that is currently off-line.
2. For record types, whose records can exist in any area, a CRS was invoked without having on-line the first (as specified in the DDL) extra area with sufficient space.
3. FFS or FNS was invoked for an off-line area.
4. All areas allowing pointer indices were not on-line for data base alteration.

****60**** Data base not available ****60****

Possible Causes:

1. This command status is returned only when multiple users are accessing the data base.
2. The user is not allowed to open the data base because some other user has opened the data base on an exclusive basis.

****61**** Exclusive open refused ****61****

Possible Causes:

1. This command status is returned only when multiple users are accessing the data base.
2. An exclusive open was not granted as some other user had the data base already open.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Statuses

****62**** Operation rejected due to active lock ****62****

Possible Causes:

1. This command status is returned only when multiple users are accessing the data base.
2. The operation requested was not performed since there was an active lock on the record or set specified.
3. The run unit is attempting to access a record when the access request, if granted, would cause a deadlock situation.
4. The command status is issued after the waiting time specified with MCC is exhausted.
5. The run unit encountering this situation can invoke NCI (and TRABT, if applicable) and restart its processing. This is not advisable if the run unit is in the midst of logging a complex transaction.

****63**** Operation rejected due to passive lock ****63****

Possible Causes:

1. The SYSTEM record cannot be actively locked because it is always the current owner of every system-owned set (i.e., it is always passively locked).
2. This command status is returned only when multiple users are accessing the data base, unless condition 1 exists.
3. The operation requested was refused as there was a passive lock on the record specified.
4. The run unit is attempting to access a record when the access request, if granted, would cause a deadlock situation.
5. The command status is issued after the waiting time specified with MCC is exhausted.
6. The run unit encountering this situation can invoke NCI (and TRABT, if applicable) and restart its processing. This is not advisable if the run unit is in the midst of logging a complex transaction.

****65**** Invalid use of CALC record ****65****

Possible Causes:

1. The CRA command was used for a record type having a calc key.

****66**** Record may not be removed from fixed set ****66****

Possible Causes:

1. The retention for the set specified has been declared as fixed in the DDL specifications.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Statuses

70 No transaction in progress **70**

Possible Causes:

1. The TRABT or TRCOM command was invoked when no complex transaction was in progress.
2. TRBGN has not been invoked.

71 Transaction already in progress **71**

Possible Causes:

1. The TRBGN command was invoked before the prior complex transaction was terminated by a TRABT or TRCOM command.

72 Posting not active **72**

Possible Causes:

1. The TRABT command was invoked without having previously invoked PIFD.
2. PIFD has not been invoked before opening the data base.

73 Posting table overflow **73**

Possible Causes:

1. This command status can be returned when posting is active.
2. The transaction is committed to the data base, even though a TRABT was attempted.
3. Consider defining several shorter transaction sequences to replace the existing transaction sequence.

74 Log file not present **74**

Possible Causes:

1. Using the RTL form of MDBS, the data base was opened without the log file being present. Processing continues.

75 Unable to re-insert record into all sets **75**

Possible Causes:

1. The value of a sort key field changed and there was insufficient memory for resorting the affected set occurrences. The record is removed from all set occurrences for which there is insufficient resorting room.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Statuses

****80**** Load error from DMSLDR ****80****

Possible Causes:

1. An attempt was made to load a non-executable file.
2. There are presently too many operating system files open to allow another to be opened.

****81**** Communications Failure ****81****

Possible Causes:

1. This command status can appear only in versions of the system that require interprocess communication (e.g., multiuser versions).
2. It indicates that an unexpected communication error was encountered during interprocess communication (e.g., a user tries to start a run unit while the data base control system is not executing).

****255**** End of Set ****255****

Command status **255** also appears as command status **-1** if the alter end of set (ALTEOS) command has run within the application program. This command status simply means the DML command tried unsuccessfully to locate a record in a set and reached the end of the set without finding the desired record. This command status can occur frequently during normal data base proocessing.

****100-199**** Catastrophic errors ****100-199****

Command status errors 100 through 199 are particular cases of a general "catastrophic" error, which would result in the integrity of the data base being destroyed. If any of these **extremely rare** command status errors is obtained, the user should contact the Micro Data Base Systems, Inc. technical support staff for aid in solving the problem. A few general notes on some of these errors are provided below.

100 Hole table inconsistency. The table which specifies how much room is on each page in this area lists more free bytes than the page itself lists. No work around.

101 Invalid record occurrence. An invalid record id was encountered when trying to determine the record type of the record occurrence. If possible, remove this record from the set.

103 Cannot find a connected record when resorting. When reordering a sorted set, problems with set connections are encountered. This could happen only for a "put" command. No work around.

104 Record not fully connected. When inserting a record into a set whose order is next or prior, a partially connected record was encountered.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Statuses

106 Invalid record id encountered during a comparison operation. When comparing two item values, an invalid record occurrence was encountered.

107 Pointer not found in higher level index (DPA) during insertion. Invalid pointer encountered when moving a DPA. Possible work around: Use Boolean commands to move all members into $SYSSET, remove all members from the original set, then reinsert the members back from $SYSSET.

108 Cannot determine item type. When trying to locate an item, the data base control system could not find an item type in the item descriptors.

109 Cannot determine record type during a "find" command. An invalid record id was found when building the sort key value.

110 Cannot find pointer in higher level during insertion. Invalid pointer encountered when splitting DPA. Possible work around: use Boolean commands to move all members into $SYSSET, remove all members from the original set, then reinsert the members back from $SYSSET.

111 Cannot find pointer in higher level during removal. Invalid pointer encountered when removing a record from a set. Possible work around: use Boolean commands to move all members into $SYSSET, remove all members from the original set, then reinsert the members back from $SYSSET.

112 SYSTEM is not owner of a set. This set is expected to be a system-owned set, but the set descriptor does not list SYSTEM as an owner.

113 Cannot find pointer in higher level during insertion. Invalid pointer encountered when updating higher DPAs after first element has changed. Possible work around: use Boolean commands to move all members into $SYSSET, remove all members from the original set, then reinsert the members back from $SYSSET.

114 Record reference not found in DPA during removal. The record for which removal was attempted is not completely connected in this set.

116 Record not found in set descriptor. When doing a "find" operation, a record was encountered that could not be found in the set descriptor for this set.

117 SYSTEM record DPA not found. When doing a "find" operation, a record was encountered that could not be found in the set descriptor for this set.

119 Invalid record type in set descriptor. When doing a "find" operation on a sorted set, a record was encountered that could not be found in the set descriptor for this set.

120 Record not found in set descriptor. When removing a record from a set, a record was encountered that could not be found in the set descriptor for this set.

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Command Statuses

121 Invalid record type encountered during Boolean operation. An invalid record id was encountered when trying to determine the record type of the record occurrence. If possible, remove the record from the set.

124 Record not found in set descriptor. When checking for duplicates, a record was encountered that could not be found in the set descriptor for this set.

126 Error in locating record. An invalid record id was encountered when trying to determine the record type of the record occurrence. If possible, remove the record from the set.

130 Master unable to deallocate user descriptor. The user descriptor was not found when processing user log-out.

© Copyright 1987 Micro Data Base Systems, Inc.

APPENDIX A

DML COMMAND FORMATS

MDBS DMS MANUAL
Appendix A

ALTEOS (no arguments) XIV–2
AMM,set–1,set–2,set–3 . . . . . XIII–3
AMO,set–1,set–2,set–3 . . . . . XIII–4
AOM,set–1,set–2,set–3 . . . . . XIII–5
AOO,set–1,set–2,set–3 . . . . . XIII–6
AUI,iblk XII–2
CCU,iblk XII–3
CRA,rec,area,iblk VII–2
CRS,rec,iblk VII–4
DBCLS (no arguments) XII–4
DBCLSA,area XII–5
DBCNV,iblk XII–6
DBENV,iblk XII–8
DBINIT,host language variable XIV–3
DBLDR, host language variables XIV–4
DBOPN,iblk XII–10
DBOPNA,area,iblk XII–12
DBSAVE (no arguments) XII–13
DBSEL,host language variable XIV–5
DBSTAT,oblk XII–14
DEFINE,blk,list XIV–6
DMSSJP,host language jump description,command status array XIV–7
DRC (no arguments) X–2
DRM,set–1 . . . . . X–3
DRO,set–1 . . . . . X–4
EXTEND,blk,list XIV–8
FDRK,rec,iblk III–3
FFM,set–1 . . . . . III–4
FFO,set–1 . . . . . III–5
FFS,area (the area argument is optional) III–6
FLM,set–1 . . . . . III–7
FLO,set–1 . . . . . III–8
FMI,item,set–1,iblk III–9
FMSK,set–1,iblk III–10
FNM,set–1 . . . . . III–11
FNMI,itm,set–1,iblk III–12
FNMSK,set–1,iblk III–13
FNO,set–1 . . . . . III–14
FNOI,itm,set–1,iblk III–15
FNOSK,set–1,iblk III–16
FNS,area (the area argument is optional) III–17
FOI,item,set–1,iblk III–18
FOSK,set–1,iblk III–19
FPM,set–1 . . . . . III–20
FPMI,itm,set–1,iblk III–21

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Appendix A

FPMSK,set−1,iblk . . . . . III−22
FPO,set−1 . . . . . III−23
FPOI,itm,set−1,iblk . . . . . III−24
FPOSK,set−1,iblk . . . . . III−25
FRK,rec,iblk . . . . . III−26
GAFN,are,oblk . . . . . XI−1
GAI,area,oblk . . . . . XI−2
GDC,oblk . . . . . XI−4
GETC,oblk . . . . . IV−3
GETM,set−1,oblk . . . . . IV−4
GETO,set−1,oblk . . . . . IV−5
GFC,itm,oblk . . . . . IV−6
GFM,itm,set−1,oblk . . . . . IV−7
GFO,itm,set−1,oblk . . . . . IV−8
GHRV,rec,itm,oblk . . . . . XI−6
GII,rec,itm,oblk . . . . . XI−7
GKI,keystr,oblk . . . . . XI−10
GLRV,rec,itm,oblk . . . . . XI−13
GMC,set−1,oblk . . . . . XII−15
GOC,set−1,oblk . . . . . XII−16
GRI,recstr,oblk . . . . . XI−14
GSI,setstr,oblk . . . . . XI−17
GTC,oblk . . . . . XII−17
GTM,set−1,oblk . . . . . XII−18
GTO,set−1,oblk . . . . . XII−19
IMS,set−1 . . . . . VIII−2
IOS,set−1 . . . . . VIII−3
LGFILE,iblk . . . . . XVI−3
LGFLSH (no arguments) . . . . . XVI−4
LGMSG,iblk . . . . . XVI−5
MAU,iblk . . . . . XV−6
MCC,iblk . . . . . XV−7
MCF (no arguments) . . . . . XV−8
MCP (no arguments) . . . . . XV−9
MPL,iblk . . . . . XIV−9
MRID,oblk . . . . . XV−10
MRTF,rec (rec is optional) . . . . . XV−11
MRTP,rec . . . . . XV−12
MSF,set−1 (set−1 is optional) . . . . . XV−13
MSP,set−1 . . . . . XV−14
NCI (no arguments) . . . . . XII−20
ODRK,rec,blk . . . . . IV−9
OFM,set−1,oblk . . . . . IV−10
OFO,set−1,oblk . . . . . IV−11
OLM,set−1,oblk . . . . . IV−12
OLO,set−1,oblk . . . . . IV−13

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Appendix A

OMI,item,set−1,blk . . . . . IV−14
OMSK,set−1,blk . . . . . IV−15
ONM,set−1,oblk . . . . . IV−16
ONMI,itm,set−1,blk . . . . . IV−17
ONMSK,set−1,blk . . . . . IV−18
ONO,set−1,oblk . . . . . IV−19
ONOI,itm,set−1,blk . . . . . IV−20
ONOSK,set−1,blk . . . . . IV−21
OOI,item,set−1,blk . . . . . IV−22
OOSK,set−1,blk . . . . . IV−23
OPM,set−1,oblk . . . . . IV−24
OPMI,itm,set−1,blk . . . . . IV−25
OPMSK,set−1,blk . . . . . IV−26
OPO,set−1,oblk . . . . . IV−27
OPOI,itm,set−1,blk . . . . . IV−28
OPOSK,set−1,blk . . . . . IV−29
ORK,rec,blk . . . . . IV−30
PFC,itm,iblk . . . . . V−3
PFM,itm,set−1,iblk . . . . . V−4
PFO,itm,set−1,iblk . . . . . V−5
PIFD,iblk . . . . . XVI−6
PUTC,iblk . . . . . V−6
PUTM,set−1,iblk . . . . . V−7
PUTO,set−1,iblk . . . . . V−8
RMS,set−1 . . . . . IX−2
ROS,set−1 . . . . . IX−3
RSM,set−1 . . . . . IX−4
RSO,set−1 . . . . . IX−5
SCD,iblk . . . . . VI−5
SCM,set−1 . . . . . VI−6
SCN (no arguments) . . . . . VI−7
SCO,set−1 . . . . . VI−8
SCU,iblk . . . . . VI−9
SDC,oblk . . . . . VI−10
SETPBF,list . . . . . XIV−10
SMC,set−1 . . . . . VI−11
SME,set−1 . . . . . VI−12
SMM,set−1,set−2 . . . . . VI−13
SMN,set−1 . . . . . VI−14
SMO,set−1,set−2 . . . . . VI−15
SMU,set−1,iblk . . . . . VI−16
SOC,set−1 . . . . . VI−17
SOE,set−1 . . . . . VI−18
SOM,set−1,set−2 . . . . . VI−19
SON,set−1 . . . . . VI−20
SOO,set−1,set−2 . . . . . VI−21

© Copyright 1987 Micro Data Base Systems, Inc.

MDBS DMS MANUAL
Appendix A

SOU,set–1,iblk VI–22
SUC,iblk VI–23
SUM,set–1,iblk VI–24
SUN,iblk VI–25
SUO,set–1,iblk VI–26
SUU,iblk VI–27
TCN (no arguments) XII–21
TCT,rec XII–22
TMN,set–1 . . . . . XII–23
TMT,rec,set–1 . . . . . XII–24
TON,set–1 . . . . . XII–25
TOT,rec,set–1 . . . . . XII–26
TRABT (no arguments) XVI–7
TRBGN (no arguments) XVI–8
TRCOM (no arguments) XVI–10
TUN,iblk XII–27
UNDEF (no arguments) XIV–11
VARCMD,host language variable XIV–12
VARCS,host language variable XIV–13
XMM,set–1,set–2,set–3 . . . . . XIII–7
XMO,set–1,set–2,set–3 . . . . . XIII–8
XOM,set–1,set–2,set–3 . . . . . XIII–9
XOO,set–1,set–2,set–3 . . . . . XIII–10

© Copyright 1987 Micro Data Base Systems, Inc.

INDEX

Active lock, XV-1
Application development, I-1
Assignment commands, VI-1, XII-9
Automatic retry, XV-2
BLF, I-3
Boolean commands, XIII-1
Buffer region size, I-3
Calc key, III-2, III-3, V-2, VII-2
Calc record, III-2
Clustered records, VII-1
Command status, I-2, II-4, II-6, III-2, IV-2, V-2, V-3, VI-2, XVII-1
Command string, II-5, XIV-1
Complex transaction, II-7, XVI-1
Connect commands, VIII-1
Create commands, VII-1
Currency indicators, I-1, II-2, V-1, VI-1, XI-1
Current member, II-2, III-1, III-4, IV-4, V-1, VI-1, VI-6
Current of run unit, II-2, III-1, IV-3, V-1, V-3, VI-1, VI-5, VIII-1
Current owner, II-2, III-1, III-5, IV-5, V-1, VI-1
Data base control system, I-1, VII-1, VIII-9, XII-8
Data base key, VI-10
Data block, II-1, II-3, II-6, IV-1, V-1, XIV-1, XIV-6
Data compression, VII-1
Data dictionary, XI-1
Data integrity, VII-1
Data Manipulation Language, I-1, I-2, III-1
Data retrieval command, IV-1
Data security, VII-1
Data transfer, I-1, II-1
Date formats, XII-6
Delete commands, X-1
Disconnect commands, VIII-9
End-of-set command status, XIV-1
Error trapping, XIV-7
Feasibility range, V-3, VII-1
Find commands, III-1, VI-3
Fixed retention, VIII-9
Host language variable, IV-1, V-1, V-3, VI-5
Host language, II-1, II-4, II-5, III-2
IDML, I-3
Log file, XVI-2
Modify commands, V-1

MDBS DMS MANUAL
Index

Multi-user environment, XII-4, XII-11
Multiuser environment, I-1, I-2, II-2, II-6, IV-1, XV-1
Multiuser locks, II-6
Page buffer region, II-7
Page buffers, XII-1, XII-4, XII-13
Page image file, II-7, XII-11, XVI-1
Passive lock, XII-8, XV-1
Pointer indices, V-2, VII-1
Program record, II-1, II-3, II-4, II-6, V-1
QRS, I-3, XV-2
RDL, I-3
Read access codes, III-2, IV-2
Record group, XIII-1
Resultant set, XIII-3
Schema access commands, XI-1
Set index, VIII-1
Set relationship, VI-3, VIII-1, VIII-9
Sort key, V-2, XII-9
Special commands, XIV-1
Transaction log, II-8
User priorities, XIV-1
User-defined currency indicators, II-3, VI-2
User-defined indicators, XII-2
Utility commands, XII-1
Write access codes, V-2

© Copyright 1987 Micro Data Base Systems, Inc.

DOCUMENTATION COMMENT FORM

MDBS Document Title:_____

We welcome and appreciate all comments and suggestions that can help us to improve our manuals and products. Use this form to express your views concerning this manual.

Please do not use this form to report system problems or to request materials, etc. System problems should be reported to MDBS by phone or telex, or in a separate letter addressed to the attention of the technical support division. Requests for published materials should be addressed to the attention of the marketing division.

Sender:

_____ _____
(name) (position)

_____ _____
(company) (telephone)

_____
(address)

_____
(city, state, zip)

COMMENTS:

Areas of comment are general presentation, format, organization, completeness, clarity, accuracy, etc. If a comment applies to a specific page or pages, please cite the page number(s).

_____

_____

_____

_____

_____

_____

_____

© COPYRIGHT 1986 Micro Data Base Systems, Inc.